РОСЖЕЛДОР

Федеральное государственное бюджетное образовательное учреждение высшего образования
«Ростовский государственный университет путей сообщения»

(ФГБОУ ВО РГУПС)

О.В. Игнатьева

ИНФОРМАТИКА И ПРОГРАММИРОВАНИЕ

Учебно-методическое пособие для лабораторных работ

Часть 2

Ростов-на-Дону
2017

УДК 681.3.06(07) + 06

Рецензент – кандидат технических наук, доцент В.В. Жуков

Игнатьева, О.В.

Информатика и программирование: учебно-методическое пособие для лабораторных работ. В 2 ч. Ч. 2 / О.В. Игнатьева; ФГБОУ ВО РГУПС. – Ростов н/Д, 2017. – 164 с.

В учебно-методическом пособии приведены задания для выполнения лабораторных работ по дисциплине «Информатика и программирование» (второй семестр) на тему «Расширенные возможности программирования на языке С++».

Предназначено для студентов и магистрантов направлений «Информатика и вычислительная техника», «Информационные системы и технологии» и «Механотроника и робототехника», изучающих дисциплины «Информатика и программирование», «Информатика», «Программирование», «Программирование на языке С++», а также для всех студентов магистратуры, бакалавриата и специалитета различных направлений, изучающих смежные дисциплины и спецкурсов.

Одобрено к изданию кафедрой «Вычислительная техника и автоматизированные системы управления».

ОГЛАВЛЕНИЕ

ЛАБОРАТОРНАЯ РАБОТА №1. УКАЗАТЕЛИ, АДРЕСА И ССЫЛКИ В С++

Цель лабораторной работы

Получить практические навыки разработки программ на языке С++ с использованием указателей и адресов, управление динамической памятью.

Методические указания

Указатели, адреса и ссылки

Указатели чаще всего используются для работы с динамической памятью и в качестве параметров функций.

Указатели – это особый вид переменных, предназначенный для хранения адресов областей памяти, выделяемых компилятором для хранения значений переменных.

Адрес переменой – это номер первого байта области памяти, отведенной для хранения значения переменой. Для переменных разного типа отводится разное количество байт, которое может быть выяснено с помощью функции **sizeof ( )**. Поэтому указатель не является самостоятельным типом, он всегда связан с каким-либо другим конкретным типом.

Формат оператора описания указателя:

тип (*имя);

Пример:

Int *f; // указатель **f** на целый тип

float *link, sum, *p;// указатели **link** и **p** на вещественный тип.

Для того, чтобы направить указатель на какую-либо переменную, используется либо операция получения адреса – символ **&** «амперсанд»:

int a = 45; f = &a; //указатель **f** содержит адрес целой переменной **a**

float c = 34.07; link = &c; /* указатель **link** содержит адрес вещественной переменной **c** ***/**

Либо используется другой уже инициализированный указатель:

p = link; /* указатель **p** теперь указывает на ту же переменную, что и указатель **link,** то есть на переменную **c***/

Значение переменной можно изменить косвенно через ее указатель:

***f=*f+2**; // аналогично **a = a+2**, то есть косвенное обращение к **a;**

В отличие от оператора смещение указателя: **f = f + 2**; при котором изменяется не значение переменной **a**, на которую указывает **f,** а изменяется адрес, который записан в **f**, то есть происходит смещение указателя на другую область памяти. Такие арифметические операции с указателями автоматически учитывают размер типа, адресуемого указателем. Значение указателя изменяется на величину **sizeof ( ) ,** умноженную на константу (в данном примере на 2). Эти операции имеют смысл в основном при работе со структурами данных, размещаемых в памяти последовательно, например, с массивами.

Ссылки

Ссылка – модифицированная форма указателя и представляет собой *синоним* имени простой переменной.

Формат оператора объявления ссылки:

тип & имя;

Для того, чтобы инициализировать ссылку, ей нужно присвоить имя простой переменной и тогда она становиться её синонимом.

Пример:

i**nt a = 3**;// описание целой переменной

int &p = a; /* описание ссылки **p** на целый тип и инициализация ссылки переменной **a** */

После такого присвоения адрес ссылки **p** будет равен адресу переменной **a** и ее значение будет равно тоже значению переменной **a**. Таким образом, ссылка становится *псевдонимом* переменной **a**. Чаще всего ссылки используются в качестве *параметров функции,* что будет рассмотрено ниже.

Указатели и массивы

В языке C/C++ имя (идентификатор) массива является указателем на его нулевой элемент. Другими словами, если, например, описан массив : i**nt a[10]**; , то его имя **a** – это тоже самое, что и **&a[0]**, а если применить операцию разадресации (*) к имени массива, то получим его нулевой элемент, то есть следующие два выражения эквивалентны: ***a** и **a[0]** также, как и: **a** и **&a[0].**

Так как элементы массива всегда располагаются в смежных областях памяти, поэтому доступ к каждому **i**-му элементу можно осуществить, смещая указатель **a** на **i** позиций. Другими словами, запись **a[i]** можно заменять на ***(a+i).** Аналогично можно описать указатель, присвоить ему адрес начала массива и работать с массивом через указатель, смещая указатель на 1, тем самым переходя к очередному элементу массива. Этот способ проиллюстрирован в следующем примере.

Но существуют и отличия массива от указателя:

- имя массива не может ни на что указывать;
- указатель, в отличие от массива, можно перенаправить на любую другую переменную такого же типа, а массив – нет, массив всегда указывает на свой нулевой элемент.

Пример:

Вычисление суммы элементов массива с использованием дополнительно-го указателя

```
#include<stdio.h>
#include<stdlib.h>
#include<stdio.h>
#include<time.h>
#include<iostream.h>
void main()
{
        int a[5]={3,18,25,7,12};
        int sum = 0, i;
        int *p;
        p = a; // аналогично p = &a[0];
        for ( i = 0; i < 5 ; i ++ )
         {      sum+=*p; // аналогично sum+=a[i] ; sum+=*(a+i) ;
                p++; // смещение указателя к следующему элементу
        }
}
```

Второй вариант того же цикла:

```
for ( i = 0; i < 5 ; i ++ )
{ sum+=p[i]; }// аналогично sum + =*(p+i); sum+=*(a+i) ;
```

Варианты заданий

Задача №1. Указатели и адреса

1. Ввести значение 2-х целых переменных *a* и b. Направить два указателя на эти переменные. С помощью указателя увеличить значение переменной *a* в 2 раза. Затем поменять местами значения переменных a и b через их указатели.
2. Ввести значение 2-х целых переменных *a* и b. Направить два указателя на эти переменные. С помощью указателя увеличить значение переменной a в 2 раза если a>b иначе b уменьшить в 2 раза
3. Ввести значение 2-х вещественных переменных a и b. Направить два указателя на эти переменные. С помощью указателя увеличить значение переменной a в 3 раза. Затем поменять местами значения переменных a и b через их указатели.
4. Ввести значение 2-х вещественных переменных a и b. Направить два указателя на эти переменные. Если a>b, то с помощью указателя увеличить значение переменной a на 3 и b уменьшить в 3 раза, в противном случае a уменьшить в 2 раза и b увеличить на 3.
5. Ввести значение 2-х символьных переменных a и b. Направить два указателя на эти переменные. С помощью указателя изменить значение переменной *a* . Затем поменять местами значения переменных a и b через их указатели.
6. Ввести значение 2-х целых переменных a и b. Направить два указателя на эти переменные. Большее из них с помощью указателя увеличить в 5 раз и меньшее уменьшить на 5.
7. Ввести значение 3-х целых переменных a и b и c. Направить указатели на эти переменные. С помощью указателя увеличить значение переменной a в 2 раза. Затем поменять местами значения переменных c и b через их указатели.
8. Ввести значение 3-х вещественных переменных a и b и c. Направить указатели на эти переменные. С помощью указателя увеличить значение переменной c в 3 раза. Затем поменять местами значения переменных a и c через их указатели.
9. Ввести значение 2-х вещественных переменных a и b. Направить два указателя на эти переменные. Большее из них с помощью указателя увеличить на7 и меньшее уменьшить на 3.
10. Ввести значение 2-х символьных переменных a и b. Направить два указателя на эти переменные. Затем поменять местами значения переменных a и b через их указатели.
11. Ввести значение 2-х целых переменных a и b. Направить два указателя на эти переменные. Затем поменять местами значения переменных a и b через их указатели.
12. Ввести значение 2-х вещественных переменных a и b. Направить два указателя на эти переменные. Затем поменять местами значения переменных a и b через их указатели.
13. Ввести значение 2-х целых переменных a и b. Направить два указателя на эти переменные. С помощью указателя увеличить значение переменной a в 2 раза, a b уменьшить в 2 раза
14. Ввести значение 2-х вещественных переменных a и b. Направить два указателя на эти переменные. С помощью указателя увеличить значение переменной a в 3 раза , a b уменьшить в 3 раза
15. Ввести значение 2-х вещественных переменных a и b. Направить два указателя на эти переменные. С помощью указателя увеличить значение переменной a в 3 раза , a b уменьшить в 3 раза

Задача №2 Указатели и Динамическая память с помощью оператора new()

1. Описать 2 указателя на целый тип. Выделить для них динамическую память. Ввести значения в выделенную память с клавиатуры. Уменьшить в 2 раза 1-ую переменную.
2. Описать 2 указателя на вещественный тип. Выделить для них динамическую память. Ввести значения в выделенную память с клавиатуры. Увеличить в 2 раза 1-ую переменную.
3. Описать 3 указателя на символьный тип. Выделить для них динамическую память. Ввести значения в выделенную память с клавиатуры.
4. Описать 2 указателя на логический тип. Выделить для них динамическую память. Присвоить значения true и false в выделенную память.
5. Описать 2 указателя на целый тип. Выделить для них динамическую память. Присвоить произвольные значения в выделенные ячейки в операторе присвоения.
6. Описать 3 указателя на вещественный тип. Выделить для них динамическую память. Присвоить произвольные значения в выделенные ячейки в операторе присвоения. Уменьшить в 2 раза 1-ую переменную.
7. Описать 1 указатель на символьный тип. Выделить для него динамическую память. Присвоить произвольное значение в выделенную ячейку в операторе присвоения.
8. Описать 2 указателя на целый тип. Выделить для них динамическую память. Ввести значения в выделенную память с клавиатуры. Поменять местами их значения.
9. Описать 2 указателя на вещественный тип. Выделить для них динамическую память. Ввести значения в выделенную память с клавиатуры. Поменять местами их значения.
10. Описать 3 указателя на символьный тип. Выделить для них динамическую память. Ввести значения в выделенную память с клавиатуры. Поменять местами значения первых 2-х переменных.
11. Описать 2 указателя на логический тип. Выделить для них динамическую память. Присвоить значения true и false в выделенную память . Поменять местами их значения.
12. Описать 2 указателя на целый тип. Выделить для них динамическую память. Присвоить произвольные значения в выделенные ячейки в операторе присвоения. Поменять местами их значения.
13. Описать 3 указателя на вещественный тип. Выделить для них динамическую память. Присвоить произвольные значения в выделенные ячейки в операторе присвоения. Поменять местами значения первых 2-х переменных.
14. Описать 1 указатель на символьный тип. Выделить для него динамическую память. Присвоить произвольное значение в выделенную ячейку в операторе присвоения.
15. Описать 1 указатель на целый тип. Выделить для него динамическую память. Ввести значения в выделенную память с клавиатуры. Затем Увеличить ее на 2.

Задача №3 Динамические массивы

1. Создать динамические массивы, используя указатели**.** Задан одномерный массив a (n). Найти количество, все номера и произведение элементов массива меньших 1.
2. Создать динамические массивы, используя указатели. Дано 2 массива x(n) и y(m). Сколько раз встречается второй элемент первого массива x(n) во втором массиве y(m).
3. Создать динамические массивы, используя указатели. В каком из двух данных массивов p (n) q(n) больше отрицательных элементов?
4. Создать динамические массивы, используя указатели. Дан массив p(n). Каждый положительный элемент в нем возвести в квадрат. Остальные элементы оставить прежними.
5. Создать динамические массивы, используя указатели. Задан одномерный массив a (n). Найти номер последнего элемента меньшего заданного числа betta, количество положительных элементов и сумму элементов больших 3.
6. Создать динамические массивы, используя указатели. В каком из двух данных массивов p (n) q(n) больше положительных элементов?
7. Создать динамические массивы, используя указатели. Дано 2 массива x(n) и y(m). Сколько раз встречается первый элемент первого массива x(n) во втором массиве y(m).
8. Создать динамические массивы, используя указатели. Дан массив r(n). Каждый элемент равный 0 в нем заменить на 1. Остальные оставить прежними.
9. Создать динамические массивы, используя указатели. Задан одномерный массив a (n). Найти номер последнего элемента равного 5 и переставить его с первым элементом массива. Найти среднее арифметическое элементов массива больших заданного числа alfa.
10. Создать динамические массивы, используя указатели. Задан одномерный массив a (n). Найти номер последнего положительного элемента и переставить его с первым элементом массива. Найти количество и сумму элементов отрицательных массива.
11. Создать динамические массивы, используя указатели. Дано 2 массива x(n) и y(m). Сколько раз встречается последний элемент первого массива x(n) во втором массиве y(m).
12. Создать динамические массивы, используя указатели. В каком из двух данных массивов p (n) q(n) больше нулевых элементов?
13. Создать динамические массивы, используя указатели. Задан одномерный массив a (n). Найти все номера и среднее арифметическое отрицательных элементов массива
14. Создать динамические массивы, используя указатели. Дано 2 массива x(n) и y(m). Сколько раз встречается второй элемент второго массива y(m) в первом массиве x(n).
15. Создать динамические массивы, используя указатели. Дано 2 массива x(n) и y(m). Сколько раз встречается первый элемент второго массива y(m) в первом массиве x(n).
16. Создать динамические массивы, используя указатели. В каком из двух данных массивов p (n) q(n) больше элементов, равных 1?

ЗАДАЧА №4

1) Создать динамические массивы, используя указатели. Дан массив b (n) . Переписать в массив C(n) положительные элементы массива b(n) умноженные на 5 (*со сжатием., без пустых элементов внутри*) Затем упорядочить методом «выбора и перестановки» по возрастанию новый массив.

2) Создать динамические массивы, используя указатели. Дан массив a (n) .Переписать в массив b(n) только положительные элементы массива a, умноженные на 3. (*со сжатием., без пустых элементов внутри*) Затем упорядочить методом «выбора и перестановки» по возрастанию новый массив.

3) Создать динамические массивы, используя указатели. Дан массив x (n) . Переписать в массив y(n) отрицательные элементы массива x умноженные на 2. (*со сжатием., без пустых элементов внутри*) Затем упорядочить методом «выбора и перестановки» по возрастанию новый массив.

4) Создать динамические массивы, используя указатели. Дан массив b (n) . Переписать в массив C(n) отрицательные элементы массива b(n) умноженные на 4. (*со сжатием., без пустых элементов внутри*) Затем упорядочить методом «выбора и перестановки» по возрастанию новый массив.

5) Создать динамические массивы, используя указатели. Дан массив b (n) . Переписать в массив C(n) положительные элементы массива b(n) деленные на 5. (*со сжатием., без пустых элементов внутри*) Затем упорядочить методом «выбора и перестановки» по возрастанию новый массив.

6) Создать динамические массивы, используя указатели. Дан массив a (n) .Переписать в массив b(n) только положительные элементы массива a, деленные на 3 (*со сжатием., без пустых элементов внутри*) Затем упорядочить методом «выбора и перестановки»по возрастанию новый массив.

7) Создать динамические массивы, используя указатели. Дан массив x (n) . Переписать в массив y(n) отрицательные элементы массива x деленные на 2. (*со сжатием., без пустых элементов внутри*) Затем упорядочить по возрастанию новый массив.

8) Создать динамические массивы, используя указатели. Дан массив b (n) . Переписать в массив C(n) отрицательные элементы массива b(n). (*со сжатием., без пустых элементов внутри*) Затем упорядочить методом «выбора и перестановки»по возрастанию новый массив.

9) Создать динамические массивы, используя указатели. Дан массив c (n) . Переписать в массив x (n) все ненулевые элементы массива умноженные на 4. (*со сжатием., без пустых элементов внутри*) Затем упорядочить методом «выбора и перестановки» по возрастанию новый массив.

10) Создать динамические массивы, используя указатели. Дан массив c (n) . Переписать в массив x (n) все ненулевые элементы массива возведенные в квадрат. (*со сжатием., без пустых элементов внутри*) Затем упорядочить методом «выбора и перестановки» по возрастанию новый массив.

11) Создать динамические массивы, используя указатели. Дан массив c (n) . Переписать в массив x (n) все ненулевые элементы массива . (*со сжатием., без пустых элементов внутри*) Затем упорядочить методом «выбора и перестановки» по возрастанию новый массив.

12) Создать динамические массивы, используя указатели. Дан массив c (n) .Переписать в массив x ненулевые элементы массива c разделенные на 5. (*со сжатием.,*

без пустых элементов внутри). Затем упорядочить методом «выбора и перестановки» по возрастанию новый массив.

13) Дан массив b (n) . Переписать в массив C(n) корни квадратные из положительных элементов массива b(n) деленные на 5. (*со сжатием., без пустых элементов внутри*) Затем упорядочить методом «выбора и перестановки» по возрастанию новый массив.

14) Создать динамические массивы, используя указатели. Дан массив b (n) . Переписать в массив C(n) корни квадратные из положительных элементов массива b(n) (*со сжатием., без пустых элементов внутри*) Затем упорядочить методом «выбора и перестановки»по возрастанию новый массив.

15) Создать динамические массивы, используя указатели. Дан массив x (n) . Переписать в массив y(n) элементы массива x, большие 3. (*со сжатием., без пустых элементов внутри*) Затем упорядочить методом «выбора и перестановки»по возрастанию новый массив.

ЛАБОРАТОРНАЯ РАБОТА №2. ДИНАМИЧЕСКИЕ МАССИВЫ

Цель лабораторной работы

Получить практические навыки разработки программ на языке С++ с использованием динамических одномерных и двумерных массивов.

Методические указания

ДИНАМИЧЕСКИЕ МАССИВЫ

Динамические массивы могут быть созданы двумя способами : либо с помощью операций **new[ ]** из языка **C++,** либо с помощью функции **malloc( )** из библиотеки языка **C**, при этом нужно указать тип и количество элементов массива.

Пример:

```
int n = 100;
float *p = new float[n];
```

Создается указатель **p** на вещественный тип , в операционной памяти выделяется непрерывная область смежных ячеек памяти для размещения 100 элементов вещественного типа, и при этом адрес начальной ячейки записывается в указатель **p**

Аналогично функция **malloc(m)** из библиотеки языка **C** выделяет непрерывную область памяти, длиной **m** байтов, поэтому для создания динамического массива из предыдущего примера необходимо записать следующие операторы:

```
int n = 100; m = sizeof (float);
float *f = (float*) malloc(n*m);
```

В переменной **m** вычислено и записано количество байт, необходимых для размещения одной переменной вещественного типа. В аргументе функции **malloc ( )** указано общее количество байт: **n*m** для размещения **n=100** элементов вещественного типа.

Доступ к элементам динамического массива осуществляется точно так же как и к статическим, так как массив и указатель – одно и то же: **p[5].**

Динамические массивы *не обнуляются* при создании.

Память, выделенная для динамического массива, после использования должна быть освобождена. Для первого способа (для операции **new[ ])** с помощью оператора **delete[ ]**, для второго способа (для функции **malloc( ))** посредством функции **free( ).**

Пример:

```
delete[ ] p; //освобождение памяти для указателя p
free ( f ); // освобождение памяти для указателя f
```

Пример: Вычислить сумму элементов динамического массива.

```
#include<malloc.h>
#include<stdlib.h>
#include<time.h>
#include<iostream.h>
void main( )
{int *p;
int sum = 0, i, n;
cout<< ''Введите длину массива'';
cin >> n;
p = (int*) malloc( n*sizeof (int));/* выделение памяти для n элементов целого типа*/
for (i = 0; i < n; i ++ ) {
```

```
                p[i] = rand ()%10 – 5 ; sum+=p[i];
}
cout<< ’’Sum=’’ << sum;
free(p); // освобождения выделенной памяти
}
```

ДВУМЕРНЫЕ ДИНАМИЧЕСКИЕ МАССИВЫ

Для создания динамического двумерного массива необходимо выполнить следующую последовательность действий:

1) **float **mass**; //описать указатель на указатель;

2) **mass = (float**) malloc(n*sizeof (float*));** /* выделить память для одномерного массива указателей на будущие строки двумерного массива, состоящего из **n** элементов, где **n** – количество строк (см. рис. 1) */

3) **for (int i = 0; i < n; i ++)**

{mass [i]= (float*) malloc(m*sizeof(float));}/* в цикле для каждого элемента массива указателей выделяется память под каждую строку двумерного массива. Причем каждая строка будет содержать **m** элементов вещественного типа, где **m** – количество столбцов (см. рис. 1) */

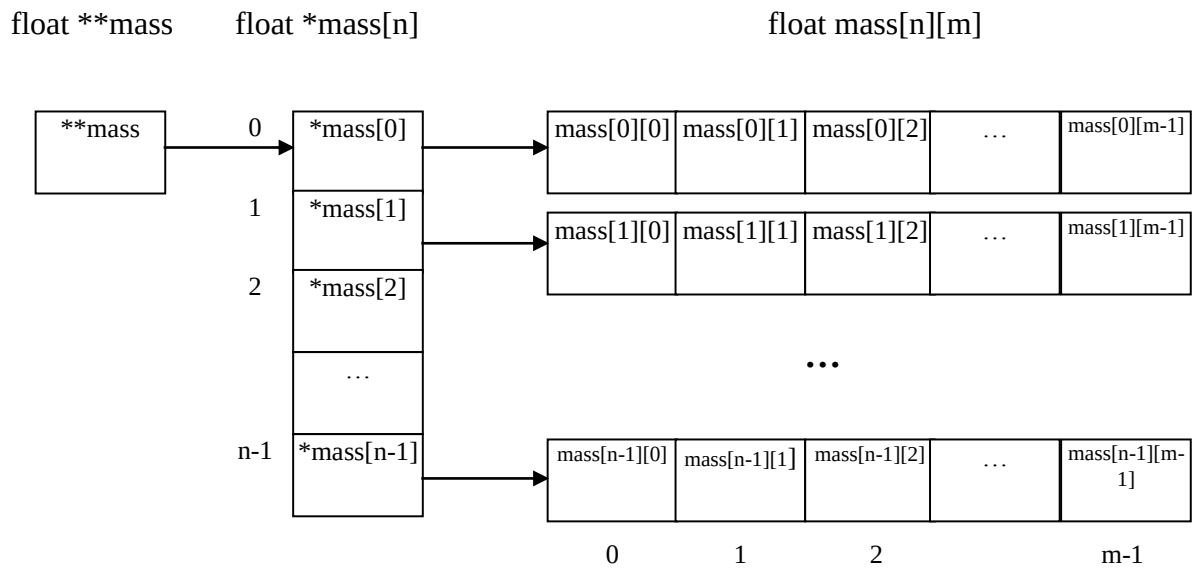


Рис. 1. « Выделение памяти для динамического двумерного массива»

Создание двумерного динамического массива с помощью средств языка С++ происходит аналогичным способом, только изменится синтаксис операторов:

```
float **mass;
mass = new float *[n];
for ( int i = 0; i < n; i ++)
{ mass[i] = new float [m]; }
```

Пример: Вычислить след матрицы, то есть произведение элементов главной диагонали матрицы с использованием двумерного динамического массива.

```
#include<stdlib.h>
#include<time.h>
#include<iostream.h>
void main ( )
{
int **matr;
int n;
cout << “Введите n”;
cin >> n;
matr = (int*) malloc (n*sizeof(int*));
 for( int i = 0; i < n ; i ++) matr[i] = (int*) malloc (n*sizeof(int));
/* заполним матрицу случайными числами */
for( int i = 0 ; i < n ; i ++)
for ( int j = 0 ; j < n ; j ++)
matr [i][j] = rand()%10;
int sled = 1;
for(i=0;i<n;i++)
{sled* = matr [i][i]; }
cout << “sled =” << sled;
}
```

Варианты заданий

Задание №1

1. Создать динамические массивы, используя указатели**.** Дан двумерный массив a, размером (n×m). Заполнить одномерный массив, найдя сумму положительных элементов в каждом столбце матрицы.
2. Создать динамические массивы, используя указатели**.** Дан двумерный массив a, размером (n×m). Заполнить одномерный массив, найдя произведение положительных элементов в каждом столбце матрицы.
3. Создать динамические массивы, используя указатели**.** Дан двумерный массив a, размером (n×m). Найти минимальный элемент в каждой строке матрицы. Затем каждую строку матрицы разделить на минимальный элемент строки.
4. Создать динамические массивы, используя указатели**.** Дан двумерный массив a, размером (n×m). Найти минимальный элемент в каждой строке матрицы среди положительных элементов.
5. Создать динамические массивы, используя указатели**.** Дан двумерный массив a, размером (n×m). Заполнить одномерный массив, найдя количество положительных элементов в каждом столбце матрицы
6. Создать динамические массивы, используя указатели**.** Дан двумерный массив a, размером (n×m). Найти минимальный элемент в каждой строке матрицы среди отрицательных элементов.
7. Создать динамические массивы, используя указатели**.** Дан двумерный массив a, размером (n×m). Заполнить одномерный массив, найдя среднее арифметическое положительных элементов в каждом столбце матрицы

8. Создать динамические массивы, используя указатели**.** Дан двумерный массив a, размером (n×m). Найти минимальный элемент в каждой строке матрицы. Затем каждую строку матрицы умножить на минимальный элемент строки.

9. Создать динамические массивы, используя указатели**.** Дан двумерный массив a, размером (n×m). Заполнить одномерный массив, найдя среднее геометрическое положительных элементов в каждом столбце матрицы

10. Создать динамические массивы, используя указатели**.** Дан двумерный массив a, размером (n×m). Найти минимальный элемент в каждой строке матрицы. Затем к каждому элементу каждой строки прибавить минимальный элемент строки.

11. Создать динамические массивы, используя указатели**.** Дан двумерный массив a, размером (n×m). Найти минимальный элемент и его номер в каждой строке матрицы. Затем из каждого элемента каждой строки вычесть номер минимального элемента строки.

12. Создать динамические массивы, используя указатели**.** Дан двумерный массив a, размером (n×m). Заполнить одномерный массив, найдя сумму отрицательных элементов в каждом столбце матрицы.

13. Создать динамические массивы, используя указатели**.** Дан двумерный массив a, размером (n×m). Найти номер минимального элемента в каждой строке матрицы. Затем к каждому элементу каждой строки прибавить номер минимального элемента строки.

14. Создать динамические массивы, используя указатели**.** Дан двумерный массив a, размером (n×m). Заполнить одномерный массив, найдя произведение отрицательных элементов в каждом столбце матрицы

15. Создать динамические массивы, используя указатели**.** Дан двумерный массив a, размером (n×m). Заполнить одномерный массив, найдя количество отрицательных элементов в каждом столбце матрицы

ЗАДАНИЕ №2

1. Создать динамические массивы, используя указатели**.** Дан двумерный массив a, размером (n×n) (или квадратная матрица a). Найти количество элементов, равных заданному числу x и расположенных в верхней треугольной матрице, расположенной выше побочной диагонали, исключая саму побочную диагональ.

2. Создать динамические массивы, используя указатели**.** Дан двумерный массив a, размером (n×n)(или квадратная матрица a). Найти номер минимального элемента её побочной диагонали.

3. Создать динамические массивы, используя указатели**.** Дан двумерный массив a, размером (n×n)(или квадратная матрица a). Найти количество и сумму отрицательных элементов, нижней треугольной матрицы, расположенной ниже побочной диагонали, исключая саму побочную диагональ.

4. Создать динамические массивы, используя указатели**.** Дан двумерный массив a, размером (n×n)(или квадратная матрица a). Найти сумму номеров минимального и максимального элементов её побочной диагонали.

5. Создать динамические массивы, используя указатели**.** Дан двумерный массив a, размером (n×n) (или квадратная матрица a). Найти количество нулевых элементов, расположенных в верхней треугольной матрице, расположенной выше побочной диагонали, включая саму побочную диагональ.

6. Создать динамические массивы, используя указатели**.** Дан двумерный массив a, размером (n×n)(или квадратная матрица a). Найти сумму номеров минимального и максимального элементов её главной диагонали.

7. Создать динамические массивы, используя указатели**.** Дан двумерный массив a, размером (n×n)(или квадратная матрица a). Найти произведение минимального и максимального элементов её главной диагонали. Затем умножить побочную диагональ на максимальный элемент главной диагонали.

8. Создать динамические массивы, используя указатели**.** Дан двумерный массив a, размером (n×n)(или квадратная матрица a). Найти среднее арифметическое положительных элементов, верхней треугольной матрицы, расположенной выше главной диагонали,

9. Создать динамические массивы, используя указатели**.** Дан двумерный массив a, размером (n×n) (или квадратная матрица a). Найти произведение элементов, расположенных в верхней треугольной матрице, расположенной выше побочной диагонали, включая саму побочную диагональ.

10. Создать динамические массивы, используя указатели**.** Дан двумерный массив a, размером (n×n)(или квадратная матрица a). Найти количество положительных элементов её главной диагонали. Затем умножить побочную диагональ на найденное количество.

11. Создать динамические массивы, используя указатели**.** Дан двумерный массив a, размером (n×n)(или квадратная матрица a). Найти среднее арифметическое положительных элементов её побочной диагонали.

12. Создать динамические массивы, используя указатели**.** Дан двумерный массив a, размером (n×n) (или квадратная матрица a). Найти произведение элементов, расположенных в верхней треугольной матрице, расположенной выше побочной диагонали, включая саму побочную диагональ.

13. Создать динамические массивы, используя указатели**.** Дан двумерный массив a, размером (n×n)(или квадратная матрица a). Найти среднее геометрическое положительных элементов её побочной диагонали.

14. Создать динамические массивы, используя указатели**.** Дан двумерный массив a, размером (n×n) (или квадратная матрица a). Найти среднее арифметическое положительных элементов, параллели главной диагонали расположенной выше над диагональю.

ЗАДАНИЕ №3

1. Создать динамические массивы, используя указатели**.** Дан двумерный массив a, размером (n×n). Упорядочить по возрастанию первую строку матрицы.

2. Создать динамические массивы, используя указатели**.** Дан двумерный массив a, размером (n×n). Упорядочить по возрастанию последнюю строку матрицы.

3. Создать динамические массивы, используя указатели**.** Дан двумерный массив a, размером (n×n). Упорядочить по возрастанию пятую строку матрицы.

4. Создать динамические массивы, используя указатели**.** Дан двумерный массив a, размером (n×n). Упорядочить по возрастанию первый столбец матрицы

5. Создать динамические массивы, используя указатели**.** Дан двумерный массив a, размером (n×n). Упорядочить по возрастанию последний столбец матрицы.

6. Создать динамические массивы, используя указатели**.** Дан двумерный массив a, размером (n×n). Упорядочить по возрастанию третий столбец матрицы

7. Создать динамические массивы, используя указатели**.** Дан двумерный массив a, размером (n×n). Упорядочить по убыванию первую строку матрицы.

8. Создать динамические массивы, используя указатели**.** Дан двумерный массив a, размером (n×n). Упорядочить по убыванию последнюю строку матрицы.

9. Создать динамические массивы, используя указатели**.** Дан двумерный массив a, размером (n×m). Найти сумму положительных элементов в каждой строке матрицы. Затем упорядочить по убыванию созданный массив

10. Создать динамические массивы, используя указатели**.** Дан двумерный массив a, размером (n×m). Найти сумму положительных элементов в каждом столбце матрицы. Затем упорядочить по убыванию созданный массив.

11. Создать динамические массивы, используя указатели**.** Дан двумерный массив a, размером (n×m). Найти среднее арифметическое положительных элементов в каждом столбце матрицы. Затем упорядочить по убыванию созданный массив.

12. Создать динамические массивы, используя указатели**.** Дан двумерный массив a, размером (n×m). Найти среднее геометрическое положительных элементов в каждой строке матрицы. Затем упорядочить по убыванию созданный массив.

13. Создать динамические массивы, используя указатели**.** Дан двумерный массив a, размером (n×m). Упорядочить по убыванию последний столбец матрицы. А также далее упорядочить по возрастанию первую строку матрицы.

14. Создать динамические массивы, используя указатели**.** Дан двумерный массив a, размером (n×m). Упорядочить по убыванию третий столбец матрицы. А также далее упорядочить по возрастанию пятую строку матрицы.

15. Создать динамические массивы, используя указатели**.** Дан двумерный массив a, размером (n×m). Переставить третий и пятый столбец. Затем упорядочить по убыванию первый столбец матрицы

ЗАДАНИЕ №4

1) Создать динамические массивы, используя указатели**.** Дан двумерный массив a, размером (n×n). Упорядочить по возрастанию главную диагональ**.**

2) Создать динамические массивы, используя указатели**.** Дан двумерный массив a, размером (n×n). Упорядочить по возрастанию параллель побочной диагонали расположенной под диагональю

3) Создать динамические массивы, используя указатели**.** Дан двумерный массив a, размером (n×n). Упорядочить по возрастанию параллель главной диагонали расположенной над диагональю.

4) Создать динамические массивы, используя указатели**.** Дан двумерный массив a, размером (n×n). Упорядочить по убыванию главную диагональ.

5) Создать динамические массивы, используя указатели**.** Дан двумерный массив a, размером (n×n). Упорядочить по убыванию параллель побочной диагонали расположенной под диагональю.

6) Создать динамические массивы, используя указатели**.** Дан двумерный массив a, размером (n×n). Упорядочить по убыванию параллель главной диагонали расположенной над диагональ

7) Создать динамические массивы, используя указатели**.** Дан двумерный массив a, размером (n×n). Упорядочить по возрастанию параллель главной диагонали расположенной под диагональю.

8) Создать динамические массивы, используя указатели**.** Дан двумерный массив a, размером (n×n). Упорядочить по возрастанию параллель побочной диагонали расположенной над диагональю.

9) Создать динамические массивы, используя указатели**.** Дан двумерный массив a, размером (n×n). Упорядочить по убыванию побочную диагональ.

10) Создать динамические массивы, используя указатели**.** Дан двумерный массив a, размером (n×n). Упорядочить по убыванию параллель главной диагонали расположенной под диагональю

11) Создать динамические массивы, используя указатели**.** Дан двумерный массив a, размером (n×n). Упорядочить по убыванию параллель побочной диагонали расположенной над диагональю.

12) Создать динамические массивы, используя указатели**.** Дан двумерный массив a, размером (n×n). Упорядочить по возрастанию побочную диагональ.

13) Создать динамические массивы, используя указатели**.** Дан двумерный массив a, размером (n×n). Упорядочить по возрастанию параллель побочной диагонали расположенной под диагональю.

14) Создать динамические массивы, используя указатели**.** Дан двумерный массив a, размером (n×n). Упорядочить по возрастанию параллель главной диагонали

15) Создать динамические массивы, используя указатели**.** Дан двумерный массив a, размером (n×n). Упорядочить по возрастанию побочную диагональ.

ЛАБОРАТОРНАЯ РАБОТА №3. СТАНДАРТНЫЙ КЛАСС C++ STRING

Цель лабораторной работы

Получить практические навыки разработки программ на языке C++ с использованием стандартного класса String.

Методические указания

СТАНДАРТНЫЙ КЛАСС C++ STRING

Как уже отмечалось ранее, в языке C+ + существует стандартный класс **string,** который предоставляет более широкие возможности для обработки строк. Для использования этого класса необходимо подключить к программе заголовочный файл <**string**>

Формат оператора описания строки:

string имя ;

Пример:

string str1, str2;

Ввод строки может происходить одним из двух способов:

1. **cin>> str1**;// ввод будет происходить до первого разделителя, в том числе до пробела
2. **getline(cin, str2**);

Вывод строки происходит как вывод единого целого:

cout <<"строка="<<str1;

ФУНКЦИИ ДЛЯ РАБОТЫ СО СТОКАМИ

Присвоение строк происходит с помощью операции присвоения:

str1 = "Mary";

str2 = str1;

Доступ к отдельным символам строки может происходить также как к элементам массива

str1[2] = 'n';// str1="Many";

Операция сцепления строк дописывает в строку еще одного слова или любую строку**;**

str2 = str1+"Hello";

Методы класса **string:**

- **S2.insert(pos, S1**) – вставляет в строку S2 строку S1, начиная с позиции pos;
- **S1.remove(pos, N)** – удаление из строки S1 N-символов, начиная с позиции pos;
- **S1.find(S2)** – поиск первого вхождения строки S2 внутри строки S1
- **S1.find(S2,pos)** – поиск первого вхождения и номера позиции(**pos**) первого вхождения строки S2 внутри строки S1.

Варианты заданий

Задание 1

1) Даны два слова. (две переменные). Сколько раз во втором слове встречается первая буква первого слова. (не использовать find, erase, substr…)

2) Даны два текста (две переменные).. Вычислить количество предложений в каждом из них. (не использовать find, erase, substr…)

3) Даны два слова по 8 символов (две переменные).. Сколько раз во втором слове встречается последняя буква первого слова. (не использовать find, erase, substr…)

4) Дан текст. Опередить в каких позициях в нем встречаются символы «;».(не использовать find, erase, substr…) (другими словами : Вывести на экран номера позиций в которых встречается символ «;»)

5) Дан текст. Переставить в нем первую букву первого слова и первую букву последнего слова . (Сначала найти номер последнего пробела). (не использовать find, erase, substr…)

6) Дан текст. Опередить в каких позициях в нем начинается каждое новое предложение (сначала найти позиции точек) (не использовать find, erase, substr…)

7) Даны два слова (две переменные). Сколько раз во втором слове встречается третья буква первого слова. (не использовать find, erase, substr…)

8) Дан текст. Переставить в нем первую букву первого предложения и первую букву последнего предложения . (Сначала найти номер последней точки без учета точки в конце всего текста). (не использовать find, erase, substr…)

9) Даны два слова (две переменные). Сколько раз в первом слове встречается третья буква второго слова. (не использовать find, erase, substr…)

10) Дан текст. Переставить в нем первую букву первого предложения и первую букву второго предложения . (Сначала найти номер первой точки). (не использовать find, erase, substr…)

11) Дан текст. Сколько раз в нем встречается символ «=».(не использовать find, erase, substr…)

12) Дан текст. Опередить в каких позициях в нем начинается каждое новое слово (сначала найти позиции пробелов) (не использовать find, erase, substr…)

13) Даны два текста (две переменные). В каком из них больше слов? При условии, что слова разделяются только одним пробелом. Сначала найти количество пробелов в каждом тексте. (не использовать find, erase, substr…)

14) Дан текст. Переставить в нем первую букву первого слова и первую букву второго слова . (Сначала найти номер первого пробела). (не использовать find, erase, substr…)

15) Даны два слова (две переменные). Сколько раз в первом слове встречается последняя буква второго слова. (не использовать find, erase, substr…)

16) Дан текст. Сколько раз в нем встречается символ «@».(не использовать find, erase, substr…)

17) Дан текст. Вывести на экран номера позиций в которых встречается символ «@».(не использовать find, erase, substr…)

Задание 2.

1. Даны 3 слова - ваши Имя, Отчество, Фамилия в 3-х разных переменных. Образовать новую символьную переменную, хранящую только ваши инициалы (через точку и пробел). (использовать склейку «+»)(не использовать find, erase, substr…)

2. Даны 3 слова в 3-х разных переменных. Образовать новую последовательность символов, состоящую из последних букв каждого слова (слитно без пробелов). (использовать склейку «+»)(не использовать find, erase, substr…)

3. Даны 3 слова в 3-х разных переменных. Образовать новую последовательность символов, состоящую из первых букв каждого слова. (слитно без пробелов). (использовать склейку «+»)(не использовать find, erase, substr…)

4. Даны 3 слова - ваши Имя, Отчество, Фамилия в 3-х разных переменных.

Образовать новую символьную переменную, хранящую полностью «имя отчество фамилию». (использовать склейку «+»)(не использовать find, erase, substr…)

5. Даны 2 слова Образовать новую символьную переменную, в которой должны чередоваться буквы первого и второго слова. использовать склейку «+»)((не использовать find, erase, substr…)

6. Вводится 3 строки - фамилия, имя и отчество учащегося. Образовать новую последовательность, оставить только фамилию и инициалы через пробел и точку .

7. Даны 4 слова в 4-х разных переменных. Образовать новую последовательность символов, состоящую из вторых букв каждого слова (слитно). (использовать склейку «+»)(не использовать find, erase, substr…)

8. Даны 3 слова в 3-х разных переменных. Образовать новую последовательность символов, состоящую из первых букв каждого слова через пробел (использовать склейку «+»)(не использовать find, erase, substr…)

9. Даны 3 слова в 3-х разных переменных. Образовать новую последовательность символов, состоящую из последних букв каждого слова (через пробел). (использовать склейку «+»)(не использовать find, erase, substr…)

10. Даны 3 слова в 3-х разных переменных. Образовать новую последовательность символов, состоящую из первых букв каждого слова. (через пробел). (использовать склейку «+»)(не использовать find, erase, substr…)

11. Даны 3 слова в 3-х разных переменных. Образовать новую символьную переменную, хранящую все три слова через пробел (использовать склейку «+»)(не использовать find, erase, substr…)

12. Даны 3 слова в 3-х разных переменных. Образовать новую символьную переменную, хранящую все три слова через запятую (использовать склейку «+»)(не использовать find, erase, substr…)

13. Даны 3 слова в 3-х разных переменных. Образовать новую символьную переменную, хранящую все три слова через запятую и пробел (использовать склейку «+»)(не использовать find, erase, substr…)

14. Даны 3 слова - ваши Имя, Отчество, Фамилия в 3-х разных переменных. Образовать новую символьную переменную, хранящую только ваши инициалы (через точку и пробел). (использовать склейку «+»)(не использовать find, erase, substr…)

Задание 3

1) Имеется некоторая последовательность символов. Образовать новую последовательность, включив в нее символы исходной, кроме символов «g» и «v». (использовать склейку «+»)(не использовать find, erase, substr…)

2) Имеется некоторая последовательность символов. Образовать новую последовательность, включив в нее символы исходной, кроме символов пробелов, точек и запятых (использовать склейку «+»)(не использовать find, erase, substr…)

3) Имеется некоторая последовательность символов. Образовать новую последовательность, включив в нее символы исходной, кроме символов пробелов. (использовать склейку «+»)(не использовать find, erase, substr…)

4) Образовать последовательность символов, включив в нее символы данной последовательности, расположенные на четных позициях. (не использовать if) (не использовать find, erase, substr…)

5) Дан текст. Переписать в другую переменную только буквы латинского алфавита и пробелы. (использовать склейку «+»)(не использовать find, erase, substr…)

6) Дан текст. Переписать в другую переменную только цифры и символы арифметических операций. (использовать склейку «+»)(не использовать find, erase,

substr…)

7) Дан текст. Переписать в другую переменную только цифры . (использовать склейку «+»)(не использовать find, erase, substr…)

8) Образовать последовательность символов, включив в нее символы данной последовательности, расположенные на нечетных позициях. (не использовать if) (использовать склейку «+»)((не использовать find, erase, substr…)

9) Имеется некоторая последовательность символов. Образовать новую последовательность, удвоив каждый символ «=» и пропустив пробелы. (использовать склейку «+»)(не использовать find, erase, substr…)

10) Имеется некоторая последовательность символов. Образовать новую последовательность, пропустив пробелы. (использовать склейку «+»)(не использовать find, erase, substr…)

11) Дан текст. Переписать в другую переменную все символы за исключением цифр и символов арифметических операций. (использовать склейку «+»)(не использовать find, erase, substr…)

12) Дан текст. Переписать в другую переменную только все символы за исключением цифр (использовать склейку «+»)(не использовать find, erase, substr…)

13) Имеется некоторая последовательность символов. Образовать новую последовательность, включив в нее символы исходной, кроме точек. (использовать склейку «+»)(не использовать find, erase, substr…)

14) Имеется некоторая последовательность символов. Образовать новую последовательность, включив в нее символы исходной, кроме запятых. (использовать склейку «+»)(не использовать find, erase, substr…)

Задание 4

1) Имеется некоторый текст. Образовать из него новый, в который включить информацию, заключенную между пробелом и запятой. (не использовать find)

2) Вводится строка - фамилия, имя и отчество учащегося. Вывести на экран преобразованную строку: оставить только фамилию и инициалы. (не использовать find)

3) Имеется некоторая последовательность символов. Образовать новую последовательность, включив в нее символы исходной в обратном порядке(не использовать find)

4) Задан текст из символов латинского алфавита, содержащий букву «а». Напечатать все символы, расположенные за первой буквой «а» до ее второго вхождения или до конца текста. (не использовать find)

5) *В* предложении, вводимом с клавиатуры в одну переменную, поменять местами первое и последнее слова. (не использовать find) *использование функции* substr( )»

6) С клавиатуры вводится слово. Определить, является ли оно «перевертышем», т.е. читается одинаково слева направо и справа налево. (сначала записать его в обратном порядке) (не использовать find)

7) Имеется некоторый текст. Образовать из него новый, в который включить информацию, заключенную между единственным пробелом и первой точкой. (не использовать find)

8) Задан текст, содержащий пару квадратных скобок. Создать новый текст, включив в него текст заключенный в квадратные скобки. (не использовать find)

9) Вводится строка - фамилия, имя и отчество учащегося. Вывести на экран преобразованную строку: оставить только фамилию и имя. (не использовать find)

10) Даны 2 слова в 2-х разных переменных одинаковой длины. Образовать новую последовательность в которой должны чередоваться буквы первого , второго слова , в обратном порядке, (не использовать find)

11) Задан текст, содержащий символ «=». Напечатать все символы, расположенные за первым вхождением «=» до его второго вхождения или до конца текста. (не использовать find)

12) В предложении, вводимом с клавиатуры в одну переменную,, поменять местами первое и последнее слова. (не использовать find) *использование функции* substr( )

13) Задан текст, содержащий пару фигурных скобок. Создать новый текст, включив в него текст заключенный в фигурные скобки. (не использовать find)

14) Задан текст, содержащий скобки. Поменять местами первое и последнее слово заключенное в скобки. (не использовать find) *использование функции* substr( )

ЛАБОРАТОРНАЯ РАБОТА №4. ОБРАБОТКА СТРОК

Цель лабораторной работы

Разработка программ языке C++ для обработки строк.

Методические указания

Сравнение строк

Как происходит сравнение строк? Происходит оно согласно специальному, ***лексикографическому порядку*** и это регламентировано стандартом.

Лексикографический порядок определяется следующим образом: если две строки имеют одинаковый размер и они посимвольно равны, значит ни одна из них не меньше другой лексикографически. Если одна строка является префиксом другой, тогда она лексикографически меньше другой. В противном случае результат определяется путём сравнением первого символа, который различен в строках.

Пример:

```
std::string obscene{"obscene"};
std::string obscenity{"obscenity"};
assert(obscene < obscenity);
assert(!(obscene < obscene) && !(obscene > obscene));
assert(!(obscenity < obscene));
```

Согласно, что “obscen**e**“ и “obscen**i**ty” имеют общий префикс “obscen”, но после этого префикса они расходятся и ‘e’ идёт в алфавите раньше ‘i’(имеет меньший ASCII код), следовательно строка “obscene“ считается меньше строки “obscenity”.

Ещё пример:

```
#include <iostream>
#include <string>
#include <algorithm>
#include <vector>

int main()
{
   std::vector<std::string> dictionary = {"obscene",
      "obscenity",
      "alphabet",
      "row",
      "column",
      "rowboat"};
   std::sort(dictionary.begin(), dictionary.end());
   std::cout << "Sorted by < operator: \n";
   for(auto str : dictionary)
      std::cout << str << "\n";
   return 0;
}
```

5

6

7

8

9

Вывод:

Sorted by < operator:
alphabet
column
obscene
obscenity
row
rowboat

Поиск в строках

Одной из самых востребованных операций над строками является операция поиска подстроки. Так как строка в C++ является контейнером, и имеет соответствующий интерфейс, то для поиска в строке, можно использовать стандартные алгоритмы C++:

- **std::find, std::find_if, std::find_if_not** – различные вариации поиска символа в строке.
- **std::adjacent_find** – поиск двух одинаковых, смежных символа.
- **std::search** – поиск первого вхождения подстроки в строку.
- **std::find_end** - поиск последнего вхождения подстроки в строку
- **std::find_first_of** – поиск одного из заданных символов в строке
- **std::search_n** – поиск *n* одинаковых символов подряд.
- **std::mismatch** – поиск несовпадения в двух строках

Кроме функций из стандартной библиотеки, string, также, содержит и свой набор методов:

- **std::string::find, std::string::rfind** – поиск символа, или подстроки в строке. Первая версия для поиска первого вхождения, а вторая для поиска последнего вхождения
- **std::string::find_first_of** – поиск первого вхождения одного из заданных символов
- **std::string::find_last_of** – поиск последнего вхождения одного из заданных символов
- **std::string::find_first_not_of** – поиск первого символа, который не представлен в заданном списке
- **std::string::find_last_not_of** – поиск последнего символа, который не представлен в заданном списке

Методы string фактически повторяют те, что уже есть в стандартной библиотеке. Иногда методы string более удобны, т.к. сделаны специально для работы со строками.

Кроме того, в отличии от обобщённых алгоритмов, методы string возвращают численную позицию символа в строке, где для не найденного символа (эквивалент end() итератора) существует специальная константа: std::string::npos.

Рассмотрим примеры применения вышеназванных функций:

Для работы примеров нам понадобятся следующие заголовки:

```
#include <iostream>
#include <string>
#include <algorithm>
#include <cassert>
```

Также, в большинстве примеров будут использованы одни и те же объекты, приведем их определение лишь один раз:

```
std::string first{"Long Sleeve the Kin"};
std::string second{"ABBA BABBAB"};
decltype((first.cbegin()) iter;
size_t position = 0;
```

Пример поиска символа в подстроке:

```
iter = std::find(first.cbegin(), first.cend(), 'n');
position = std::distance(first.cbegin(), iter);
std::cout << "First 'n' position is: "<< position << "\n";
assert(position == first.find('n'));
```

Вывод:

```
1       First 'n' position is: 2
```

L	**o**	**n**	**g**		**S**	**l**	**e**	**e**	**v**	**e**		**t**	**h**	**e**		**K**	**i**	**n**
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18

В примере выше, использовали как std::find, так и std::string::find и их результат, безусловно, идентичен. Кроме того, запись вызова обобщённого алгоритма уступает специализированному в лаконичности.

Теперь рассмотрим поиск с условием:

```
iter = std::find_if(first.cbegin(), first.cend(), [](char symbol){return symbol > 'p'; });
position = std::distance(first.cbegin(), iter);
std::cout << "First greater than 'p' symbol's position is: "<< position << "\n";
iter = std::find_if_not(first.cbegin(), first.cend(),
   [](char symbol){return symbol > 'p'; });
position = std::distance(first.cbegin(), iter);
std::cout << "First lesser or equal to 'p' symbol's position is: "<< position << "\n";
```

Вывод:

?

```
1       First greater than 'p' symbol's position is: 9
2       First lesser or equal to 'p' symbol's position is: 0
```

L	**o**	**n**	**g**		**S**	**l**	**e**	**e**	**v**	**e**		**t**	**h**	**e**		**K**	**i**	**n**
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18

В примере выше мы использовали одну и ту же лямбда-функция для поиска разных символов, в первом случае мы искали первый символ, чей ASCII код будет больше чем ‘p‘. Во втором случае мы искали первый символ с кодом меньше. Подобный поиск можно выполнить только с применением обобщённых алгоритмов, т.к. сам класс string не содержит ничего подобного. Это, в целом, довольно легко объяснить – условный поиск по строке мне видится несколько надуманным.

Рассмотрим поиск одинаковых, смежных символов:

```
iter = std::adjacent_find(first.cbegin(), first.cend());
position = std::distance(first.cbegin(), iter);
std::cout << "Position of the first symbol in an adjacent pair is: "<< position << "\n";
```

Вывод:

?

```
1	Position of the first symbol in an adjacent pair is: 7
```

L	o	n	g		S	l	e	e	v	e		t	h	e		K	i	n
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18

Подобной функции в самом классе string тоже нет.

Поиск подстроки в строке:

?

```
1	std::string eve{"eve"};
2	iter = std::search(first.cbegin(), first.cend(), eve.cbegin(), eve.cend());
3	position = std::distance(first.cbegin(), iter);
4	std::cout << R"(Position of the "eve" substring is: )"<< position << "\n";
5	assert(position == first.find("eve"));
```

Вывод:

?

```
1	Position of the "eve" substring is: 8
```

L	o	n	g		S	l	e	e	v	e		t	h	e		K	i	n
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18

Согласно примеру, вызов обобщенной функции **search** несколько перегружен. Кроме того, мы не можем просто передать строку в качестве параметра функции – нам необходимо создавать строку отдельно и передавать искомую подстроку через итераторы, что неудобно. А всё из-за того, что алгоритм **search** создан для последовательностей, а в общем случае последовательность, конечно же, будет представлена итераторами. С другой стороны, у класса string есть собственный метод для поиска подстроки – **string::find**. Мы уже видели его ранее, когда искали символ. Как можно заметить использование **string::find** гораздо удобнее – нам не надо думать об итераторах, мы просто передаём ту строку, что нам надо найти. Более того, использование string::find может быть более эффективно за счёт того, что **string::find** имеет перегруженную версию, которая принимает указатель на литерал, что позволяет не создавать лишний **string**.

Теперь поищем *последнее* вхождение заданной подстроки. Конечно, мы могли бы использовать **search + std::reverse_iterator**, но есть отдельная функция для этого.

?

```
std::string bb{"BB"};
iter = std::search(second.cbegin(), second.cend(), bb.cbegin(), bb.cend());
position = std::distance(second.cbegin(), iter);
std::cout << R"(Position of the first "BB" substring is: )"<< position << "\n";
assert(position == second.find("BB"));
iter = std::find_end(second.cbegin(), second.cend(), bb.cbegin(), bb.cend());
position = std::distance(second.cbegin(), iter);
std::cout << R"(Position of the last "BB" substring is: )"<< position << "\n";
assert(position == second.rfind("BB")););
```

Вывод:

```
?
1  Position of the first "BB" substring is: 1
2  Position of the last "BB" substring is: 7
```

A	B	B	A		B	A	B	B	A	B
0	1	2	3	4	5	6	7	8	9	10

Первоначально мы нашли первую позицию **BB**, чтобы лучше увидеть разницу – функции действительно нашли разные позиции. Кроме того, мы использовали string::rfind, для получения того же самого результата, но, как и раньше, в гораздо более лаконичной форме.

Далее переходим к поиску позиции первого символа, который входит(или не входит) в заданное множество символов:

```
?
std::string symbols{"g K"};
iter = std::find_first_of(first.cbegin(), first.cend(), symbols.cbegin(),
symbols.cend());
position = std::distance(first.cbegin(), iter);
std::cout << R"(Position of the first symbol from "g K" set is: )"<< position
<< "\n";
assert(position == first.find_first_of(symbols));
position = first.find_first_not_of("gnolL");
std::cout << R"(Position of the first symbol different from "gnolL" is: )"
   << position << "\n";
```

Вывод:

```
?
1  Position of the first symbol from "g K" set is: 3
2  Position of the first symbol different from "gnolL" is: 4
```

L	o	n	g		S	l	e	e	v	e		t	h	e		K	i	n
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18

Как и ранее, специализированная версия поиска символа более лаконичная. Более того, тогда как в **string** есть версия, которая позволяет найти первый отличный от множества символ, обобщённого алгоритма для этого случая нет. Конечно, вы всегда можете использовать **std::find_first_of** передав туда предикат, но это сделает его запись ещё более громоздкой.

Найдя первую позицию, найдём и последнюю:

```
?
iter = std::find_first_of(first.crbegin(), first.crend(), symbols.cbegin(),
```

```
    symbols.cend()).base();
position = std::distance(first.cbegin(), iter) - 1;
std::cout << R"(Position of the last symbol from "g K" set is: )"<< position
<< "\n";
assert(position == first.find_last_of(symbols));
position = first.find_last_not_of("niK ");
std::cout << R"(Position of the last symbol different from "niK " is: )"
  << position << "\n";
```

Вывод:

?

```
1  Position of the last symbol from "g K" set is: 16
2  Position of the last symbol different from "niK " is: 14
```

L	o	n	g		S	l	e	e	v	e		t	h	e		K	i	n
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18

В отличии от string, обобщённого алгоритма для поиска последнего символа из множества не существует, поэтому нам пришлось использовать обратные итераторы. Как и в предыдущем примере, если бы мы захотели использовать обобщённый алгоритм для поиска последней позиции символа отличного от символов множества, нам бы пришлось добавить предикат.

Теперь найдём последовательность заданной длины, состоящую из одинаковых символов.

?

```
iter = std::search_n(first.cbegin(), first.cend(), 2, 'e');
position = std::distance(first.cbegin(), iter);
std::cout << "Position of the first symbol in sequence of 2 'e' is: " << position
<< "\n";
assert(position == first.find(std::string(2, 'e')));
```

Вывод:

?

```
1  Position of the first symbol in sequence of 2 'e' is: 7
```

L	o	n	g		S	l	e	e	v	e		t	h	e		K	i	n
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18

В классе string операция поиска последовательности одинаковых символов в строке отсутствует. С другой стороны это довольно легко решается созданием временной строки; правда, это делает решение с использованием метода string менее эффективным.

Рассмотрим пример поиска расхождения между двумя строками:

?

```
std::string trueStatement{"ABBA is the most famous Sweden band."};
std::string falseStatement{"ABBA is the most famous American band."};
auto ret = std::mismatch(trueStatement.begin(), trueStatement.end(),
  falseStatement.begin());
assert(ret.first != trueStatement.end());
assert(ret.second != falseStatement.end());
size_t posInFirst = std::distance(trueStatement.begin(), ret.first);
```

```
size_t posInSecond = std::distance(falseStatement.begin(), ret.second);
assert(posInFirst == posInSecond);
std::cout << "Position of the first symbol in mismatch is: "<< posInFirst << "\n";
```

Вывод:

?

```
Position of the first symbol in mismatch is: 24
```

Так как мы используем строки одинаковый длины, которые имеют одинаковое начало, то и позиция где они расходятся будет идентичной. Поиск расхождения двух строк может быть осуществлён только с помощью обобщённого алгоритма, так как метода для такой операции в классе string нет.

Исходя из вышеописанных примеров и их скромного анализа, можно составить следующую таблицу соответствия методов string, обобщённым алгоритмам:

Стандартные алгоритмы	Методы string
find	find
find_if	X
find_if_not	X
adjacent_find	X
search	find
find_end	rfind
find_first_of	find_first_of
search_n	find
mismatch	X

Разделение и изменение строки

Хотя различные операции поиска весьма важны при работе со строками, они, всё же, редко применяются в изоляции. К примеру, когда мы ищем позиции некоторого символа, или строки в другой строке, следующим шагом, зачастую, является использование найденной позиции для выделения части строки в отдельную. Для этих целей в классе string есть целых 2 операции:

- **string::copy** – выделение подстроки в стиле C – копирует подстроку в буфер.
- **string::substr** – выделение подстроки в стиле C++ – возвращает подстроку в виде объекта string.

?

```
#include <iostream>
#include <string>

int main()
{
   std::string chemicals{"Argentum Mercury Ferrum"};
   char ferrum[7]{};
   auto mercury = chemicals.substr(chemicals.find('M'), 7);
   chemicals.copy(ferrum, 6, chemicals.find('F'));
   std::cout << "C++ string: " << mercury << "\n";
0  std::cout << "C string: " << ferrum << "\n";
```

```
    return 0;
}
```

Вывод:

```
C++ string: Mercury
C string: Ferrum
```

Заметьте, что порядок аргументов *{позиция, число символов}*, в **copy** и **substr** разный.

Ещё одна операция, которая довольно часто встречается при работе со строками это операция замены. Для этих целей в string существует метод **replace**. Но, к сожалению, он крайне не самостоятелен и, следовательно, крайне не удобен. Вместо того, чтобы иметь сигнатуру подобную *(что_заменить, на_что_заменить)* этот метод представлен целым набором сигнатур, где *что_заменить* представлено либо итераторами, либо позициями, но не строкой! Поэтому, вместо того, чтобы написать:

```
std::string chemicals{"Argentum Mercury Ferrum"};
chemicals.replace("Argentum ", "");
chemicals.replace(" Ferrum", ", Freddy");
```

мы вынуждены писать следующий код:

```
#include <iostream>
#include <string>

int main()
{
   std::string chemicals{"Argentum Mercury Ferrum"};
   auto mPos = chemicals.find('M');
   chemicals.replace(chemicals.begin(), chemicals.begin() + mPos, "");
   auto fPos = chemicals.find('F');
   chemicals.replace(chemicals.begin() + fPos - 1, chemicals.end(), ", Freddie");
   std::cout << "New string: " << chemicals << "\n";
   return 0;
}
```

Вывод:

```
New string: Mercury, Freddie
```

Конечно, мы могли бы также уложиться в две строчки, но они стали бы ещё длиннее и непонятнее. Кроме того, даже те строки с **replace,** что присутствуют сейчас в коде, заставляют остановиться и задуматься, тогда как “идеальный” синтаксис поглощается без остановок. Более того, с помощью **replace** мы можем заменить только одно вхождение некой строки(символа), ни о каком “заменить всё”, без явного цикла, и речи быть не может.

Другой операцией, которая также встречается очень часто, является соединение(конкатенация) строк. Для этого в классе **string** существует целый букет методов:

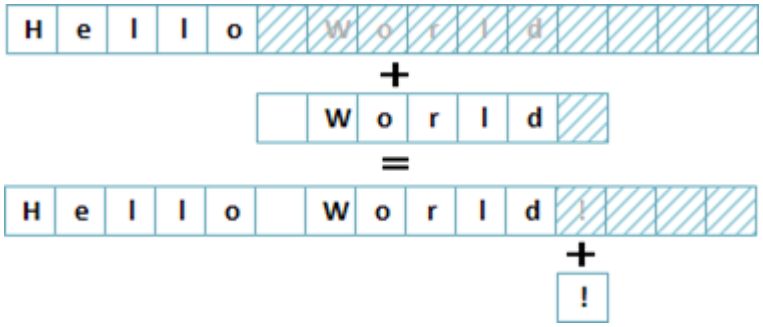
- **operator+=** – добавляет строку(символ) к концу строки.
- **operator+** – соединяет две строки, возвращая третью
- **append** – добавляет строку(символ) к концу строки. Метод похож на **operator+=,** но даёт чуть больше свободы
- **push_back** – добавляет символ в конец строки.
- **insert** – вставляет строку(символ) в произвольную позицию в строке.

Все вышеперечисленные методы, которые работают со строкой, могут добавлять как С-строки, так и С++-строки. Возможность соединения **string** с С-строкой позволяет присоединять литералы, без создания временных объектов класса **string.**

Рассмотрим пример, в котором, в частности, рассмотрим и то как выполняется оптимизация описанная ранее. Условимся, что наша реализация класса string использует оптимизацию для коротких строк, а также выделяет буфер размера больше, чем требуется для содержания текущей строки.

?

```
std::string helloWorld{"Hello"};
char exclamation{'!'};
helloWorld += " World";
helloWorld.push_back(exclamation);
```



Итак, все три слияния прошли – символы из приклеиваемых строк просто были скопированы внутрь строки. Но что если бы мы добавили ещё одну строку:

?

```
helloWorld += " Wow!";
```

Т.к. данная строка, длиною в 6 символов не может поместиться в наш существующий внутри-строковый буфер, то новый буфер необходим. Поэтому данная операция слияния уже не будет столь простой, как те, что мы видели дальше. Алгоритм сложения будет следующим:

1. Рассчитать размер минимального буфера, который необходим для хранения новой строки.
2. [*Оптимизация*] Добавить к этому размеру некое число, чтобы уменьшить количество перераспределений памяти в дальнейшем.
3. Выделить буфер необходимого размера в куче.
4. Скопировать все данные из внутреннего буфера в новый, выделенный на куче, буфер.
5. Поместить указатель на новый буфер в объединение, затерев часть данных, что мы там хранили. Эти данные нам больше не нужны.
6. Скопировать символы из строки “ Wow!” в наш новый буфер.

Как вы видите, вместо одного шага(шестого в этом списке) нам необходимо выполнить шесть(!). Об этом стоит помнить, если вы довольно интенсивно используете слияние строк.

Еще интереснее ситуация с operator+, к примеру:

?

```
std::string helloWorld{"Hello"};
helloWorld = helloWorld + " World" + "!";
```

Вот что происходит в этом коде:

1. **helloWorld + " World**" – результатом выражения будет временный объект string, назовём его **tempStr**
2. **tempStr** + "!" – результатом выражения будет временный объект string, назовём его tempStr2
3. **helloWorld = tempStr2** – поместим результирующую строку в наш объект helloWorld.

Так вот, до C++11 **tempStr** и **tempStr2** были разными строками, которые создавались в теле **operator+**, а потом удалялись. Слово "поместим", в 3-м шаге можно было бы заменить на "скопируем" для C++ кода, предшествующего новому стандарту. Таким образом для такого простого случая мы имели: создание 2-х объектов **string** и одно копирование.

Сейчас же ситуация изменилась в лучшую сторону,- благодаря семантике перемещения **tempStr и tempStr2** будут представлены одной и той же строкой, а слово "поместим" из шага 3, мы можем заменить на "переместим", что будет означать просто перемещение указателя на буфер временной строки в helloWorld. Выигрыш благодаря новому стандарту может быть весьма ощутим, т.к. в идеальной ситуации нам нужен только один буфер на сколь угодно длинное выражение содержащие соединение строк. Конечно, мир не идеален и в силу того, что размер буфера временной строки может быть в любой момент превышен, то и сохранить один буфер на длинных выражениях будет сложнее. Тут уже на ведущие роли выходит стратегия выделения памяти для буфера – от этого зависит как часто придётся перераспределять память.

Конвертируем строки

Еще одной весьма популярной операцией, при работе со строками, является преобразование строки в число и числа в строку. До C++11 в стандартной библиотеке не было никаких средств для совершения подобных преобразований, которые бы работали с объектами класса string. С появлением нового стандарта эта ситуация изменилась и мы получили следующие функции для преобразования строки в число:

- **std::stoi, std::stol, std::stoll** – строка в знаковое целое
- **std::stoul, std::stoull** – строка в беззнаковое целое
- **std::stof, std::stod, std::stold** – строка в вещественное число

И одну функцию для преобразования числа в строку: std::to_string. Почему в одну сторону мы имеем много функций, тогда как обратное преобразование обходится одной? Ответ прост – в C++ перегрузка разрешена только по аргументам функции, но не по её возвращаемому значению. Это позволяет иметь много перегруженных функций с одним именем – to_string.

Пример:

?

```
#include <iostream>
#include <string>
```

```
int main()
{
   std::string one{"1"};
   std::string fifteen{"F"};
   std::string fourteen{"1110"};
   double real = 12.567890;
   std::cout << "One to integer: " << std::stoi(one) << "\n";
   std::cout << "Fifteen to integer: " << std::stol(fifteen, nullptr, 16) << "\n";
   std::cout << "Fourteen to integer: " << std::stoull(fourteen, nullptr, 2) << "\n";
   std::cout << "Real to string: " << std::to_string(real) << "\n";
   return 0;
}
```

Вывод:

?

```
One to integer: 1
Fifteen to integer: 15
Fourteen to integer: 14
Real to string: 12.567890
```

До того, как в стандарте появились новые функции для преобразований между строкой и числом весьма популярной была функция из ***boost*** **– lexical_cast**. Она легка в использовании и, как утверждается, быстрее чем любой другой стандартный подход. Но, с появлением новых функций, **lexical_cast** более не представляет интереса, так как новые функции не менее удобны. Более того, в отличии от **lexical_cast**, новые функции могут учитывать систему счисления преобразовываемого числа, в чём мы убедились ранее.

?

```
#include <iostream>
#include <string>
#include <vector>
#include <chrono>
#include <boost/lexical_cast.hpp>

int main()
{
   std::vector<std::string> strings;
   for(size_t i = 0; i < 1000000; ++i)
      strings.push_back(std::to_string(i));

   size_t sum = 0;
   auto start = std::chrono::high_resolution_clock::now();
   for(auto& str : strings)
      sum += std::stoul(str);
   auto stdElapsed = (std::chrono::high_resolution_clock::now() - start).count();
   start = std::chrono::high_resolution_clock::now();
   for(auto& str : strings)
      sum += boost::lexical_cast<unsigned long>(str);
```

```
    auto boostElapsed = (std::chrono::high_resolution_clock::now() - start).count();

    std::cout << "Standard way took: " << stdElapsed << "\n";
    std::cout << "Boost way took: " << boostElapsed << "\n";
    float ratio = std::max<float>(stdElapsed, boostElapsed)/
      std::min<float>(stdElapsed, boostElapsed);
    std::string relation = stdElapsed < boostElapsed ?
      std::string("faster") : std::string("slower");
    std::cout << "Standard way is " << ratio << " " << relation << "!\n";
    return 0;
}
```

Его вывод:

?

```
Standard way took: 450124
Boost way took: 1070272
Standard way is 2.37773 faster!
```

В среднем было получено двукратное превосходство стандартного подхода(MSVC 2013 Update3).

Варианты заданий

Задача 1 «Найти и заменить»

в задании можно заменить русские слова на английские или сделать транслитерацию русских слов

1. Пользователь вводит текст . Вычислить количество слов начинающих на «м». Количество слов «Компьютер» или «компьютер», а также количество предложений.
2. Пользователь вводит текст . Заменить в тексте слова «ПК» на «компьютер», подсчитав их количество.
3. Пользователь вводит текст . Вывести исходный текст, заменив в нем слово «Иванов И.И.» на «Сидоров А.А.». Заменить круглые скобки на фигурные, подсчитав их количество.
4. Пользователь вводит текст. Вывести исходный текст, заменив в нем слово «Pascal» на «C++». подсчитав их количество. Вычислить количество слов «компьютер»,
5. Пользователь вводит текст . Вывести исходный текст, заменив в нем слово «плохо» на «хорошо». Вычислить количество всех слов.
6. Пользователь вводит текст . Вычислить количество слов начинающих на «А». Количество слов «мало» или «Мало». Заменить в тексте слова «доллар» на «рубль».
7. Пользователь вводит текст . Заменить в тексте слова «Максимальный» на «Наибольший». Удалить все слова «Иванов И.И.». вычислить количество предложений
8. Пользователь вводит текст . Заменить в тексте слова «кризис» на «проблема», подсчитав их количество. Удалить все слова «компьютер»,
9. Пользователь вводит текст . Вывести исходный текст, заменив в нем квадратные скобки на круглые. Вычислить количество всех слов и количество появления слова «обучаемый».
10. Пользователь вводит текст на . Вывести исходный текст, заменив в нем слово «проблема» на «задача». Удалить все слова «Иванов И.И.».

11. Пользователь вводит текст. Вывести исходный текст, заменив в нем слово «три» на «удовлетворительно». Вычислить количество слов начинающихся на «к».

12. Пользователь вводит текст . Вывести исходный текст, заменив в нем слово «дублирование» на «копирование». Вычислить количество всех слов.

13. Пользователь вводит текст . Вывести исходный текст, заменив в нем слово «четыре» на «хорошо». Вычислить количество всех слов и предложений. Заменить все скобки на пробелы.

14. Пользователь вводит текст. Вывести исходный текст, заменив в нем слово «Pascal» на «C++». Удалить символы '*'. Заменить все цифра на пробелы.

15. Пользователь вводит текст . Вывести исходный текст, заменив в нем слово «дублирование» на «копирование». Вычислить количество всех слов.

Задача 2 «Исправление ошибок в тексте»

1. Исходный текст набран с ошибками: иногда отсутствуют пробелы после точек. Вставить 1 пробел после каждой точки, если он отсутствует перед следующим предложением. А также вычислить количество слов и предложений. (не использовать find)

2. Исходный текст набран с ошибками: .Вывести исходный текст, заменив в нем строчные (малые) буквы, следующие за точкой и произвольным количеством пробелов на прописные (большие) буквы. В исходном тексте может быть много предложений и точек. А также вычислить количество слов и предложений. (не использовать find)

3. В тексте заменить цифры на их словесные названия (использовать Case) (не использовать find)

4. Исходный текст набран с ошибками: между некоторыми словами по несколько пробелов. Заменить в тексте подряд идущие пробелы одним пробелом. (не использовать find)

5. В тексте заменить символы арифметических операций(+-*/) на их словесные названия (использовать Case) (не использовать find)

6. Исходный текст набран с ошибками: после слова может находиться один или более пробелов перед точкой (или нет). Вывести исходный текст, убрав в нем эти пробелы перед точкой (между словом и точкой). В исходном тексте может быть много предложений и точек. А также удалить символы '#'.(не использовать find)

7. Исходный текст набран с ошибками . Вывести исходный текст, заменив в нем строчные(малые) буквы, следующие за точкой и одним пробелом на прописные(большие) буквы. В исходном тексте может быть много предложений и точек. Вычислить количество слов начинающихся на букву «А». (не использовать find)

8. Пользователь вводит текст на русском языке. Вывести исходный текст, заменив в нем все заглавные (прописные(большие)) буквы на строчные(малые). Вычислить количество слов а также количество предложений. А также удалить символы '@'.(не использовать find)

9. Исходный текст набран с ошибками: Выражения, заключенные в скобки имеют один пробел вначале и в конце. Вывести исходный текст, убрав в нем пробелы после открывающейся скобки, а также перед закрывающейся скобкой . В исходном тексте может быть много выражений заключенных в скобки. (не использовать find)

10. Исходный текст набран с ошибками . Вывести исходный текст, заменив в нем строчные(малые) буквы, следующие за точкой и произвольным количеством

пробелов на прописные(большие) буквы. В исходном тексте может быть много предложений и точек. А также вычислить количество слов «ПК». (не использовать find)

11. Исходный текст набран с ошибками: после слова может находиться один или более пробелов перед запятой (или нет). Вывести исходный текст, убрав в нем пробелы перед запятой (между словом и запятой). В исходном тексте может быть много предложений и запятых. А также удалить символы ‘-'.(не использовать find)

12. Исходный текст набран с ошибками: Выражения, заключенные в скобки имеют один или более пробелов вначале и в конце (или нет). Вывести исходный текст, убрав в нем пробелы после открывающейся скобки, а также перед закрывающейся скобкой . В исходном тексте может быть много выражений заключенных в скобки (не использовать find)

13. Исходный текст набран с ошибками: после слова может находиться один пробел перед запятой или нет. убрать в тексте эти пробелы перед запятой (между словом и запятой). В исходном тексте может быть много предложений и запятых. А также удалить символы '$'.(не использовать find)

14. Исходный текст набран с ошибками: некоторые слова по ошибке начинаются не с одной первой заглавной буквы, а с двух заглавных букв. Исправить текст. (не использовать find)

15. Исходный текст набран с ошибками: иногда отсутствуют пробелы после запятых. Вставить 1 пробел после каждой запятой, если он отсутствует перед следующим словом. А также вычислить количество слов «Информатика». (не использовать find)

16. Исходный текст набран с ошибками: иногда отсутствуют пробелы после точек. Вставить 1 пробел после каждой точки, если он отсутствует перед следующим предложением. а также вычислить количество предложений. А также удалить квадратные скобки. (не использовать find)

Задача № 3 «Сортировка строк»

1. Дана строка текста. Упорядочить по возрастанию символы, расположенные между первым элементом равным ‘?’ и последним элементом равным ‘!”. Предварительно найти, где они находятся.

2. Дана строка текста. Известно, что в ней есть один символ = ‘&’, найти где он находится и отсортировать по возрастанию символы в строке , расположенные за ним.

3. Дана строка текста. Известно, что в ней есть один элемент равный ‘a’, найти где он находится и упорядочить по алфавиту элементы , расположенные за этим элементом = ‘a’.

4. Дана строка текста.. Известно, что в ней есть один символ ‘*’, найти где он находится и отсортировать по возрастанию элементы массива , расположенные перед ним.

5. Дана строка текста. Найти позицию, где находиться последняя точка. Упорядочить по возрастанию символы массива расположенные перед этой точкой.

6. Дана строка текста. Отсортировать последние 7 символов в строке по алфавиту (или по таблице ASCII).

7. Дана строка текста (цифры и буквы). Известно, что в ней есть единственная цифра. Найти номер позиции, где находится цифра. Далее отсортировать буквы по алфавиту, которые расположены после этой цифры.

8. Дана строка текста. Известно, что в ней есть цифры и буквы. Переписать в другую строку только цифры и затем отсортировать ее по возрастанию.

9. Дана строка текста. Известно, что в ней есть один элемент равный ‘*’ и один элемент равный ‘#’. Найти где они находятся, и упорядочить по алфавиту символы, расположенные между ними.

10. Дана строка текста.. Известно, что в строке есть только две точки . Найти их порядковые номера. Далее упорядочить по возрастанию символы, расположенные между ними.

11. Дана строка текста.. Найти номер первого элемента равного ‘+’. Упорядочить по убыванию элементы массива расположенные за этим элементом.

12. Дана строка текста. Отсортировать символы по алфавиту с 4 позиции по 15 позицию в строке.

13. Дана строка текста. Известно, что в ней есть только буквы (прописные и срочные). Переписать во вторую строку только заглавные (прописные) буквы, и затем отсортировать их по алфавиту.

14. Даны две строки текста. Объединить их , записав обе строки в третью строку (неважно в каком порядке). Затем отсортировать третью строку по алфавиту (или по таблице ASCII).

15. Дана строка текста. Отсортировать первые 7 символов в строке по алфавиту (или по таблице ASCII).

Задача 4 «Сравнение строк»

1. Даны две строки текста. Определить сколько раз встречается каждый символ первой строки во второй строке. *Например*: Пусть исходная строка Str1 := “xyz”; Str2: “x a d c x y x w” . Тогда “x” – встречается 3 раза “y”- встречается 1 раз, “z”- встречается 0 раз.

2. Определить, можно ли путем перестановок символов в строке S1 получить строку S2. Считать, что обе строки одной длинны. Далее удалить все слова «неудача».

3. Дана строка текста. Определить сколько раз встречается каждый символ в строке. Например: Пусть исходная строка Str: “x w x y x w” . Тогда “x” – встречается 3 раза “y”- встречается 1 раз, “w”- встречается 2 раза. Далее удалить все слова «проблема».

4. Даны две строки текста. Определить сколько раз встречается каждый символ первой строки во второй строке. Например: Str1 : “xyz”; Str2: “x a d c x y x w” . Тогда “x” – встречается 3 раза “y”- встречается 1 раз, “z”- встречается 0 раз. Далее заменить все слова «компьютер» на слова «ПК» .

5. Определить, можно ли путем перестановок символов в строке S1 получить строку S2. Считать, что обе строки одной длинны. Далее вычислить количество пробелов во второй строке. Далее заменить все слова «уникальный» на слова «единственный» .

6. Даны две строки текста. Определить встречается ли хотя бы один раз каждый из символов первой строки во второй строке. Например: Пусть исходная строка Str1 : “xyz”; Str2: “x a d c x y xw” . Тогда “x” – встречается 3 раза “y”- встречается 1 раз, “z”- встречается 0 раз. Не каждый символ встречается , например “z” не встречается ни разу.

7. Дана строка текста. Определить сколько раз встречается каждый символ в строке. Например: Пусть исходная строка Str: “x w x y x w” . Тогда “x” – встречается 3

раза “y”- встречается 1 раз, “w”- встречается 2 раза. Далее вычислить количество пробелов в строке.

8. Определить, можно ли путем перестановок символов в строке S1 получить строку S2. Считать, что обе строки одной длинны. Далее вычислить количество пробелов во второй строке.

9. Дана строка текста. Определить сколько раз встречается каждый символ в строке. Например: Пусть исходная строка Str: “x w x y x w” . Тогда “x” – встречается 3 раза “y”- встречается 1 раз, “w”- встречается 2 раза.

10. Определить, можно ли путем перестановок символов в строке S1 получить строку S2. Считать, что обе строки одной длинны.

11. Даны две строки текста. Определить встречается ли хотя бы один раз каждый из символов первой строки во второй строке. Например: Пусть исходная строка Str1 : “xyz”; Str2: “x a d c x y xw” . Тогда “x” – встречается 3 раза “y”- встречается 1 раз, “z”- встречается 0 раз. Не каждый символ встречается , например “z” не встречается ни разу. Далее вычислить количество символов «а» в первой строке.

12. Даны две строки текста. Определить сколько раз встречается каждый символ первой строки во второй строке. Например: Пусть исходная строка Str1 : “xyz”; Str2: “x a d c x y x w” . Тогда “x” – встречается 3 раза “y”- встречается 1 раз, “z”- встречается 0 раз . Далее удалить все слова «кризис».

13. Даны две строки текста. Определить встречается ли хотя бы один раз каждый из символов первой строки во второй строке. Например: Пусть исходная строка Str1 : “xyz”; Str2: “x a d c x y xw” . Тогда “x” – встречается 3 раза “y”- встречается 1 раз, “z”- встречается 0 раз. Не каждый символ встречается , например “z” не встречается ни разу. Далее удалить все слова «разрушение» в первой строке.

14. Даны две строки текста. Определить сколько раз встречается каждый символ первой строки во второй строке. Например: Пусть исходная строка Str1 : “xyz”; Str2: “x a d c x y x w” . Тогда “x” – встречается 3 раза “y”- встречается 1 раз, “z”- встречается 0 раз. Далее вычислить количество пробелов во второй строке.

15. Дана строка текста. Определить сколько раз встречается каждый символ в строке. Например: Str: “x w x y x w” . Тогда “x” – встречается 3 раза “y”- встречается 1 раз, “w”- встречается 2 раза. Далее заменить все слова «уникальный» на слова «единственный» .

16. Даны две строки текста. Определить встречается ли хотя бы один раз каждый из символов первой строки во второй строке. Например: Пусть исходная строка Str1 : “xyz”; Str2: “x a d c x y xw” . Тогда “x” – встречается 3 раза “y”- встречается 1 раз, “z”- встречается 0 раз. Не каждый символ встречается , например “z” не встречается ни разу. Далее заменить все слова «компьютер» на слова «ПК» .

ЛАБОРАТОРНАЯ РАБОТА №5. МАССИВЫ СТРОК

Цель лабораторной работы

Разработка программ языке C++ с использованием массива строк.

Методические указания

В современном стандарте C++ определен класс с функциями и свойствами (переменными) для организации работы со строками (в классическом языке С строк как таковых нет, есть лишь массивы символов char):

```
#include <string>
```

Для работы со строками также нужно подключить стандартный namespace:

```
using namespace std;
```

В противном случае придётся везде указывать описатель класса **std::string** вместо string.

Ниже приводится пример программы, работающей со **string** (в старых си-совместимых компиляторах не работает!):

```
#include <iostream>
#include <string>
#include <malloc.h>
using namespace std;

int main () {
 string s = "Test";
 s.insert (1,"!");
 cout << s.c_str() << endl;
 string *s2 = new string("Hello");
 s2->erase(s2->end());
 cout << s2->c_str();
 cin.get(); return 0;
}
```

Основные возможности, которыми обладает класс string:

- **инициализация массивом символов** (строкой встроенного типа) или другим объектом типа **string**. Встроенный тип не обладает второй возможностью;
- копирование одной строки в другую. Для встроенного типа приходится использовать функцию **strcpy();**
- **доступ к отдельным символам строки для чтения и записи**. Во встроенном массиве для этого применяется операция взятия индекса или косвенная адресация с помощью указателя;
- **сравнение двух строк на равенство**. Для встроенного типа используются функции семейства **strcmp();**
- **конкатенация (сцепление) двух строк**, дающая результат либо как третью строку, либо вместо одной из исходных. Для встроенного типа применяется функция strcat(), однако чтобы получить результат в новой строке, необходимо

последовательно задействовать функции **strcpy() и strcat(),** а также позаботиться о выделении памяти;

- **встроенные средства определения длины строки** (функции-члены класса **size()** и **length()).** Узнать длину строки встроенного типа можно только вычислением с помощью функции **strlen();**
- **возможность узнать, пуста ли строка**.

Рассмотрим эти базовые возможности более подробно.

Инициализация строк при описании и **длина строки** (не включая завершающий нуль-терминатор):

```
string st( "Моя строка\n" );
cout << "Длина " << st << ": " << st.size()
    << " символов, включая символ новой строки\n";
```

Строка может быть задана и пустой:

```
string st2;
```

Для проверки того, **пуста ли строка**, можно сравнить ее длину с 0:

```
if ( ! st.size() ) // пустая
```

или применить метод **empty(),** возвращающий true для пустой строки и false для непустой:

```
if ( st.empty() ) // пустая
```

Третья форма создания строки инициализирует объект типа **string** другим объектом того же типа:

```
string st3( st );
```

Строка **st3** инициализируется строкой **st**. Как мы можем убедиться, что эти **строки совпадают**? Воспользуемся оператором сравнения (==):

```
if ( st == st3 ) // инициализация сработала
```

Как **скопировать одну строку в другую**? С помощью обычной операции присваивания:

```
st2 = st3; // копируем st3 в st2
```

Для **сцепления строк** используется операция сложения (+) или операция сложения с присваиванием (+=). Пусть даны две строки:

```
string s1( "hello, " );
string s2( "world\n" );
```

Мы можем получить третью строку, состоящую из конкатенации первых двух, таким образом:

```
string s3 = s1 + s2;
```

Если же мы хотим добавить **s2** в конец **s1**, мы должны написать:

```
s1 += s2;
```

Операция сложения может сцеплять объекты класса **string** не только между собой, но и со строками встроенного типа. Можно переписать пример, приведенный выше, так, чтобы специальные символы и знаки препинания представлялись встроенным типом **char *,** а значимые слова – объектами класса **string**:

```
const char *pc = ", ";
string s1( "hello" );
string s2( "world" );
string s3 = s1 + pc + s2 + "\n";
cout << endl << s3;
```

Подобные выражения работают потому, что компилятор "знает", как автоматически преобразовывать объекты встроенного типа в объекты класса string. Возможно и простое присваивание встроенной строки объекту string:

```
string s1;
const char *pc = "a character array";
s1 = pc; // правильно
```

Обратное преобразование при этом **не работает**. Попытка выполнить следующую инициализацию строки встроенного типа вызовет ошибку компиляции:

```
char *str = s1; // ошибка компиляции
```

Чтобы осуществить такое преобразование, необходимо явно вызвать функцию-член с названием c_str() ("строка Си"):

```
const char *str = s1.c_str();
```

Функция **c_str()** возвращает указатель на символьный массив, содержащий строку объекта string в том виде, в каком она находилась бы во встроенном строковом типе. Ключевое слово const здесь предотвращает "опасную" в современных визуальных средах возможность непосредственной модификации содержимого объекта через указатель.

К **отдельным символам** объекта **типа string**, как и встроенного типа, можно обращаться с помощью операции взятия индекса. Вот, например, фрагмент кода, заменяющего все точки символами подчеркивания:

```
string str( "www.disney.com" );
int size = str.size();
for ( int i = 0; i < size; i++ )
 if ( str[i] == '.' ) str[ i ] = '_';
cout << str;
```

Но лучше читать документацию по C++ и пользоваться его возможностями. Например, предыдущее действие мы могли бы выполнить вызовом одной-единственной функции **replace():**

```
replace( str.begin(), str.end(), '.', '_' );
```

Правда, здесь использован не метод **replace** класса **string**, а одноимённый алгоритм:

```
#include <algorithm>
```

Поскольку **объект string** ведет себя как контейнер, к нему могут применяться и другие алгоритмы. Это позволяет решать задачи, не решаемые напрямую функциями класса **string.**

Ниже приводится краткое описание основных операторов и функций класса **string**, ссылки в таблице ведут к русскоязычным описаниям в интернете. Более полный список возможностей класса string можно получить, например, в Википедии или на сайте cplusplus.com.

Задание символов в строке	
operator=	присваивает значения строке
assign	назначает символы строке
Доступ к отдельным символам	
at	получение указанного символа с проверкой выхода индекса за границы
operator[]	получение указанного символа
front	получение первого символа
back	получение последнего символа
data	возвращает указатель на первый символ строки
c_str	возвращает немодифицируемый массив символовC, содержащий символы строки
Проверка на вместимость строки	
empty	проверяет, является ли строка пустой
size **length**	возвращает количество символов в строке
max_size	возвращает максимальное количество символов
reserve	резервирует место под хранение
Операции над строкой	
clear	очищает содержимое строки
insert	вставка символов
erase	удаление символов
push_back	добавление символа в конец строки
pop_back	удаляет последний символ
append	добавляет символы в конец строки
operator+=	добавляет символы в конец строки
compare	сравнивает две строки

replace	заменяет каждое вхождение указанного символа
substr	возвращает подстроку
copy	копирует символы
resize	изменяет количество хранимых символов
swap	обменивает содержимое
Поиск в строке	
find	поиск символов в строке
rfind	поиск последнего вхождения подстроки
find_first_of	поиск первого вхождения символов
find_first_not_of	найти первое вхождение отсутствия символов
find_last_of	найти последнее вхождение символов
find_last_not_of	найти последнее вхождение отсутствия символов

Следует обратить внимание, что у любой функции класса string может быть несколько *перегрузок* - разновидностей с одинаковыми именами, отличающихся между собой списками и типами аргументов.

В качестве недостатков класса string можно отметить следующее:

- отсутствие в классе встроенных средств для разбора строк по набору разделителей (аналога функции strtok для строк char *);
- возможное замедление быстродействия по отношению к char * при сложной обработке данных.

Варианты заданий

Задание 1

1) Задан список из десяти городов. *(массив [.] string)* Подсчитать количество названий, которые оканчиваются буквой «В».
2) Задан список из десяти городов. *(массив [.] string).* Найти порядковые номера городов, в названии которых вторая буква «о».
3) Задан список из десяти городов. *(массив [.] string)*Поменять местами название последнего города таблицы и последнего города, начинающегося с буквы «к».
4) Задан список из десяти городов. *(массив [.] string).* Найти количество городов, название которых заканчивается сочетанием букв «град» или “grad”.
5) Задан список из десяти городов. *(массив [.] string)).* Найти порядковые номера городов, начинающихся с буквы «Н».
6) Задан список из десяти городов. *(массив [.] string)* Подсчитать количество названий, которые начинаются на букву «А».
7) Задан список из десяти городов. *(массив [.] string)* Найти порядковые номера городов, которые оканчиваются буквой «к».
8) Задан список из десяти городов. *(массив [.] string)*Поменять местами названия любых двух городов, заканчивающихся буквой «а».
9) Задан список из десяти городов. *(массив [.] string)*Поменять местами названия любых двух городов, начинающихся с буквы «а».
10) Задан список из десяти городов, *(массив [.] string)*поменять местами название первого города таблицы и последнего города, начинающегося с буквы «К».

11) Задан список из десяти городов. *(массив [.] string).* Подсчитать количество названий, городов, которое содержит более 4-х букв.

12) Задан список из десяти городов. *(массив [.] string).* Найти порядковые номера городов, в названии которых по 7 букв.

13) Задан список из десяти городов. *(массив [.] string).* Присвоить переменной ***st*** название последнего из городов, которое содержит более 4-х букв.

14) Задан список из десяти городов. *(массив [.] string)* Поменять местами названия первого города и любого другого, которое содержит более семи букв.

15) Задан список из десяти городов. *(массив [.] string).* Найти все порядковые номера городов, название которых заканчивается сочетанием букв «бург» или “burg”.

16) Задан список из десяти городов. *(массив [.] string)* Поменять местами названия двух городов, названия которых оканчиваются сочетанием букв «ск» или ‘sk’

Задание 2

1. Задан список из 5 имен девочек. *(массив [.] string).* Присвоить переменной d имя с наименьшим числом букв.
2. Задан список из десяти городов. *(массив [.] string).* Найти порядковый номер города, в названии которого минимальное число букв.
3. Задан список из десяти городов. *(массив [.] string).* Присвоить переменной g название города с максимальным числом букв.
4. Задан список из десяти городов. *(массив [.] string).* Поменять местами названия самого длинного и самого короткого слова.
5. Задан список из 10 имен . *(массив [.] string).* Найти количество букв в самом длинном имени.
6. Задан список из 20 названий горных вершин. *(массив [.] string).* Присвоить переменной ***st*** самое короткое название
7. Задан список из десяти городов. *(массив [.] string).* Найти порядковый номер города, в названии которого максимальное число букв.
8. Задан список из 20 названий горных вершин. *(массив [.] string).* Поменять местами названия самого длинного и самого короткого слова.
9. Задан список из 10 имен . *(массив [.] string).* Найти порядковый номер имени, в названии которого максимальное число букв.
10. Задан список из 20 названий горных вершин. *(массив [.] string).* Найти порядковый номер вершины, в названии которой максимальное число букв.
11. Задан список из 20 фамилий. *(массив [.] string).* Присвоить переменной ***st*** самую длинную фамилию
12. Задан список из 20 фамилий. *(массив [.] string).* Найти порядковый номер фамилии, в которой минимальное число букв.
13. Задан список из 20 фамилий. *(массив [.] string).* Найти количество букв в самой длиной фамилии.
14. Задан список из 20 фамилий. *(массив [.] string).* Поменять местами названия самого длинного и самого короткого слова.
15. Задан список из 20 названий горных вершин. *(массив [.] string).* Найти количество букв в самом коротком названии.

Задание 3 Задан список из десяти городов. *(массив [.] string).* Подсчитать количество названий, в которых есть буква «D». (не использовать find)

1. Задан список из десяти городов. *(массив [.] string).* Подсчитать количество названий, в которых есть по две буквы «а».

2. Задан список из десяти городов. *(массив [.] string)* . Найти номера городов в названии которых, есть по две буквы «с».

3. Задан список из десяти городов. *(массив [.] string.)*. Найти название города с максимальным количеством букв «g».

4. Задан список из десяти городов. *(массив [.] string)*. Подсчитать количество названий, в которых есть ровно по 3 буквы «о».

5. Задан список из десяти городов. *(массив [.] string)*. Подсчитать количество названий, в которых нет буквы «р». (не использовать find)

6. Задан список из 10 имен девочек. *(массив [.] string)*. Подсчитать количество имен, в которых есть хотя бы 1 буква «р». (не использовать find)

7. Задан список из 10 имен девочек. *(массив [.] string)*. Подсчитать количество имен, в которых есть ровно 1 буква «а».

8. Задан список из 10 имен девочек. *(массив [.] string)*. Найти номера имен в названии которых, есть по две буквы «а».

9. Задан список из 10 имен девочек. *(массив [.] string)*. Найти имя с максимальным количеством букв «а».

10. Задан список из десяти городов. *(массив [.] string)*. Найти номер последнего города в списке в названии, в которого есть хотя бы одна буква «t» (не использовать find)

11. Задан список из 10 имен девочек. *(массив [.] string)*. Найти номера имен, в названии которых, есть не менее двух букв «Н».

12. Задан список из десяти городов. *(массив [.] string)* . Найти номера городов в названии которых, есть не менее 3-х букв «о».

13. Задан список из десяти городов. *(массив [.] string)*. Найти номер последнего города в списке в названии, в которого есть более одной буквы «н» (не использовать find)

14. Задан список из 20 названий горных вершин. *(массив [.] string)*. Найти название с максимальным количеством букв «k».

Задача № 4

1. Дан список фамилий сотрудников. *(массив [.] string)*. Переписать в другой список только фамилии, чья длина больше 7 букв. Затем упорядочить по алфавиту второй список.

2. Дан список фамилий сотрудников. *(массив [.] string)*. Переписать в другой список только те фамилии, которые заканчиваются на ‘а’. Затем упорядочить по алфавиту второй список.

3. Дан список фамилий сотрудников. *(массив [.] string)*. Переписать в другой список только те фамилии, которые заканчиваются на ‘в’. Затем упорядочить по алфавиту второй список методом «пузырька».

4. Дан список фамилий сотрудников. Переписать в другой список только те фамилии, в которых вторая буква ‘л’. Затем упорядочить по алфавиту второй список методом «пузырька».

5. Дан список из 10 городов. *(массив [.] string)*. Переписать в другой список только те города, в которых третья буква ‘к’. Затем упорядочить по алфавиту второй список методом «пузырька».

6. Дан список из 10 городов. *(массив [.] string)*. Переписать в другой список только те города, которые заканчиваются на ‘в’. Затем упорядочить по алфавиту второй список

7. Дан список из 10 городов. Переписать в другой список только те города, чье название длиннее 7 букв. Затем упорядочить по алфавиту второй список.

8. Задан список из 20 названий горных вершин. *(массив [.] string).* Переписать в другой список только те вершины, чье название длиннее 7 букв. Затем упорядочить по алфавиту второй список.

9. Задан список из 20 названий горных вершин. *(массив [.] string).* Переписать в другой список только те вершины, название которых оканчивается на «тау» или “tau” . Затем упорядочить по алфавиту второй список.

10. Задан список из 20 названий горных вершин. *(массив [.] string).* Переписать в другой список только те вершины, название которых оканчивается на «рок» или ‘rok’. Затем упорядочить по алфавиту второй список.

11. Задан список из 20 названий горных вершин. *(массив [.] string).* Переписать в другой список только те вершины, название которых вторая буква «М». Затем упорядочить по алфавиту второй список.

12. Задан список из 10 имен девочек. *(массив [.] string).* Переписать в другой список только те имена, в которых есть ровно 1 буква «Р». Затем упорядочить по алфавиту второй список.

13. Задан список из 10 имен девочек. *(массив [.] string).* Переписать в другой список только те имена, в которых нет буквы «Р». Затем упорядочить по алфавиту второй список.

14. Дан список фамилий сотрудников. *(массив [.] string).* Переписать в другой список только те фамилии, которые заканчиваются на ‘ова’ или ‘ova’. Затем упорядочить по алфавиту второй список.

ЛАБОРАТОРНАЯ РАБОТА №6. СТРУКТУРЫ

Цель лабораторной работы

Разработка программ языке С++ с использованием структур.

Методические указания

ТИПЫ ДАННЫХ, ОПРЕДЕЛЯЕМЫЕ ПОЛЬЗОВАТЕЛЕМ

На основании простых типов данных, массивов и указателей можно построить тип данных, имеющих более сложную структуру.

1 ПЕРЕИМЕНОВАНИЕ ТИПОВ (TYPEDEF)

Для того чтобы сделать программу более ясной, можно задать типу новое имя с помощью служебного слова **typedef**.

Формат оператора переименования типа:

typedef тип новое_ имя_типа [размерность];

Пример:

typedef double real;

После такого описания нового типа его можно использовать для описания переменных, как и имена стандартных типов.

Пример:

real a,b,c;

2 ПЕРЕЧИСЛИМЫЕ ТИПЫ

Перечислимые типы описываются для переменных, принимающих только определенные значения из заданного набора. Они описываются с помощью служебного слова **enum**.

Пример:

enum day={sun, mon, tues, weds, thur, fri, sat};

По умолчанию элементы в перечислении нумеруются с 0. Поэтому, если описать две переменные d1, d2 созданным перечислимым типом **day**:

enum day d1,d2;

эти две переменные могут принимать значения от 0 до 6, и у каждого значения будет свое имя: **d1 = 2; (tues),d2 = 0 ;(sun).** Другими словами в этом случае объявлена 7 именованных констант :

```
const sun = 0;
const mon = 1;
const tues = 2;
…
const sat = 6;
```

Покажем, как можно их использовать в следующем примере.

```
enum day={sun, mon, tues, weds, thur, fri, sat};
enum day d1,d2
cin >> d1 >> d2;
switch (d1)
{ case sun: case sat: cout << " выходные дни";break;
case mon: cout << " понедельник – трудный день "; break;
default : cout << " Ошибка ";
```

Можно не по умолчанию задавать значения элементам перечисления, явно указывать значения в перечислимом типе. Это происходит следующим образом:

enum status {success = 1,wait = -1,error = 0};

Далее описываются переменные этого типа**:**

enum status Proc1_Status, Proc2 _Status;

При необходимости можно описать перечисление как новый тип и использовать его в дальнейшем без слова **enum**.

typedef enum status { success = 1, wait = -1, error = 0} name_status;

Тогда тип **name_status –** новое имя созданного типа и его можно использовать для описания переменных без слова **enum** следующим образом:

name_status Proc1_Status, Proc2_Status;

3 СТРУКТУРЫ

В отличие от массива, все элементы которого однотипны, структура может содержать элементы разных типов. В языке С++ структура является видом класса и обладает всеми свойствами.

Описание *структурного шаблона* предшествует описанию структурной переменной. Формат оператора описания структурного шаблона:

```
struct имя_структурного_шаблона
{
        тип_1 имя_элемента_1;
        тип_2 имя_элемента_2;
        ...
        тип_N имя_элемента_N;
};
```

Пример:

```
struct Automobile
{       int year; // год выпуска автомобиля
        int doors; //количество дверей автомобиля
        double horse_Power; // мощность двигателя (лошадиные силы)
        char model [10]; //название модели
};
```

Описание структурной переменной происходит следующим образом:

struct Automobile my_Car, yur_Car;

Аналогично описание структурного массива или указателя на структуру:

struct Automobile Car[10]; // массив для информации о 10 машинах;

struct Automobile *taxi; //указатель на структуру;

Для упрощения описания структурных переменных можно ввести новый тип с помощью оператора **typedef**.

```
typedef struct Automobile{
        int year; doors;
        double horse_Power;
        char model [10];
        } At_Mble;
```

Тогда далее описание структурной переменной упрощается до следующего кода:

At_Mble my_Car, yur_Car, Car[10], *taxi;

Инициализация структурной переменной происходит путем перечисления значений элементов структурной переменной в фигурных скобках в порядке их описания:

struct Automobile my_Car = {2000, 4, 100, ”BMV”}

Для *доступа к отдельным элементам (полям) структуры* используется операция выбора (.) «точка».

Пример 1:

```
my_Car . year = 2000;
my_Car . doors = 4;
my_Car . horse_Power = 100;
my_Car . model = ”BMV”;
```

Пример 2:

```
struct Automobile Car[10];
Car [i] . year = 1997;
Car [i] . doors = 5;
Car [i] . horse_Power = 200;
Car [i]. model = ”Ford”;
```

Если в программе используется *указатель на структуру, то для доступа к отдельным элементам (полям)* структуры вместо оператора выбора (.) «точка» используется оператор – > «*стрелк*а».

```
At_Mble *taxi;
taxi =( At_Mble*) malloc(n*sizeof(At_Mble));
taxi – > doors = 5;
taxi – > horse_Power = 200;
taxi – > model = ”Mersedes”;
```

СТРУКТУРЫ В КАЧЕСТВЕ ПАРАМЕТРОВ ФУНКЦИИ

Функции могут иметь параметры структурных типов, передаваемые по значению, по ссылке или по указателю. Приведем пример функции, в которую передается структура по значению, вычисляющая возраст указанной машины

Пример:

```
int Age_Car (At_Mble X_Car) // функция, вычисляющая
{int s; s = 2008 - X_Car.year; // возраст машины на текущий 2008 год
return (s);}

void main()
{At_Mble my_Car = {2000, 4, 100, ”BMV”}; /*инициализация структурной переменной*/
int Age_my_Car;
Age_my_Car = Age_Car (my_Car); //вызов функции Age_Car
cout << “ возраст моей машины= “ << Age_my_Car ;
}
```

Приведем пример функции, в которую передается структурная переменная по ссылке.

Пример:

```
void Pw(At_Mble &X_Car) { X_Car. horse_Power = random(100)+100;)
void main (){… Pw(my_Car); ...}
```

Приведем пример функции, вычисляющей сумму мощностей, и в которую передается структурный массив (точнее указатель на его нулевой элемент).

Пример:

```
double Sum_Power(At_Mble *X_Car)
{ double sum = 0;
 for (int i; i < n; i + +)
sum+ = X_Car[i]. horse_Power;
/* либо второй вариант: sum+ = X_Car– > horse_Power; X_Car+ +;*/
return (sum);
}

void main (){
At_Mble Car[10]; …
double S = Sum_Power(Car); ...
}
```

Варианты заданий

Задача №1

1. Определить комбинированный (структурный) тип для представления анкеты ребенка, состоящей из его имени, пола и роста. Ввести информацию по 20 детям. Вывести средний рост мальчиков
2. Составить программу, выводящую на экран ведомость начисленной заработной платы (Ф.И.О., должность, год и дата рождения, заработная плата). Найти среднюю зарплату. И вывести фамилии с зарплатой выше средней.
3. Определить комбинированный (структурный) тип для представления информации по горным вершинам, состоящей из названия вершины и ее высоты. Ввести информацию по 20 вершинам. Вывести среднее значение высот всех 20 вершин. Далее вывести названия всех вершин ниже среднего.
4. Составить программу, выводящую на экран ведомость начисленной заработной платы (Ф.И.О., должность, год и дата рождения, заработная плата). Вывести фамилию сотрудника с самой маленькой зарплатой.
5. Составить программу, выводящую на экран ведомость начисленной заработной платы (Ф.И.О., должность, год и дата рождения, заработная плата). Вывести фамилию сотрудника с самой большой зарплатой.
6. Определить комбинированный (структурный) тип для представления анкеты ребенка, состоящей из его имени, пола и роста. Ввести информацию по 20 детям. Вывести имя самой высокой девочки.
7. Определить комбинированный (структурный) тип для представления информации по горным вершинам, состоящей из названия вершины и ее высоты. Ввести информацию по 50 вершинам. Вывести название самой низкой вершины из всех 50.
8. Определить комбинированный (структурный) тип для представления анкеты ребенка, состоящей из его имени, пола и роста. Ввести информацию по 20 детям. Вывести средний рост девочек.
9. Определить комбинированный (структурный) тип для представления информации по горным вершинам, состоящей из названия вершины и ее высоты. Ввести информацию по 20 вершинам. Вывести среднее значение высот всех 20 вершин. Далее вывести названия всех вершин выше среднего.

10. Определить комбинированный (структурный) тип для представления анкеты ребенка, состоящей из его имени, пола и роста. Ввести информацию по 20 детям. Вывести средний рост всех детей.

11. Определить комбинированный (структурный) тип для представления информации по горным вершинам, состоящей из названия вершины и ее высоты. Ввести информацию по 10 вершинам. Вывести название самой высокой вершины из всех 10.

12. Определить комбинированный (структурный) тип для представления анкеты ребенка, состоящей из его имени, пола и роста. Ввести информацию по 20 детям. Вывести средний рост девочек. Далее вывести имена всех девочек выше среднего.

13. Определить комбинированный (структурный) тип для представления информации по горным вершинам, состоящей из названия вершины и ее высоты. Ввести информацию по 20 вершинам. Вывести среднее значение высот всех 20 вершин.

14. Определить комбинированный (структурный) тип для представления анкеты ребенка, состоящей из его имени, пола и роста. Ввести информацию по 20 детям. Вывести имя самого высокого мальчика. Вывести средний рост мальчиков. Далее вывести имена всех мальчиков ниже среднего.

15. Составить программу, выводящую на экран ведомость начисленной заработной платы (Ф.И.О., должность, год и дата рождения, заработная плата). Вывести все фамилии, начинающиеся на букву «А» и их зарплату .

16. Составить программу, выводящую на экран ведомость начисленной заработной платы (Ф.И.О., должность, год и дата рождения, заработная плата). Вывести дату рождения сотрудника с самой маленькой зарплатой.

Задача №2

1. Определить комбинированный (структурный) тип, описывающий окружность и состоящий из двух полей: «радиус» и «центр». Поле «центр» в свою очередь состоит еще из двух полей: «координата X» и «координата Y». Ввести информацию по 10 окружностям. Вывести координаты центра окружности, чей радиус самой большой.

2. Определить комбинированный (структурный) тип для представления анкеты жителя, состоящей из его фамилии, названия города, где он проживает, и городского адреса. Адрес состоит из полей: «улица», «дом», «квартира». Ввести информацию по 100 жителям. Вывести фамилии двух любых жителей, которые «По Иронии Судьбы» живут в разных городах, но по одинаковому адресу.

3. Определить комбинированный (структурный) тип для представления анкеты студента, состоящей из его фамилии, дня рождения и пола. «День рождения» состоит из полей: «число», «месяц», «год». Ввести информацию по 25 студентам из группы. Вывести фамилию самого старшего мальчика из группы.

4. Определить комбинированный (структурный) тип, описывающий окружность и состоящий из двух полей: «радиус» и «центр». Поле «центр» в свою очередь состоит еще из двух полей: «координата X» и «координата Y». Ввести информацию по 10 окружностям. Вывести координаты центра окружности, чей радиус самой маленький

5. Определить комбинированный (структурный) тип для представления анкеты жителя, состоящей из его фамилии, названия города, где он проживает, и городского адреса. Адрес состоит из полей: «улица», «дом», «квартира». Ввести информацию по 100 жителям. Вывести фамилии жителей, которые живут в одном городе с первым жителем из списка.

6. Определить комбинированный (структурный) тип для представления анкеты студента, состоящей из его фамилии, дня рождения и пола. «День рождения» состоит из полей: «число», «месяц», «год». Ввести информацию по 25 студентам из группы. Вывести все фамилии девочек, родившихся в декабре.

7. Определить комбинированный (структурный) тип, описывающий окружность и состоящий из двух полей: «радиус» и «центр». Поле «центр» в свою очередь состоит еще из двух полей: «координата X» и «координата Y». Ввести информацию по 10 окружностям. Вывести радиус окружности, чей центр самый удаленный от начала координат.

8. Определить комбинированный (структурный) тип для представления анкеты жителя, состоящей из его фамилии, названия города, где он проживает, и городского адреса. Адрес состоит из полей: «улица», «дом», «квартира». Ввести информацию по 100 жителям. Вывести фамилии жителей, которые живут в Ростове-на-Дону на улице Ленина.

9. Определить комбинированный (структурный) тип для представления анкеты студента, состоящей из его фамилии, дня рождения и пола. «День рождения» состоит из полей: «число», «месяц», «год». Ввести информацию по 25 студентам из группы. Вывести все фамилии мальчиков, родившихся в мае 1986 года.

10. Определить комбинированный (структурный) тип, описывающий окружность и состоящий из двух полей: «радиус» и «центр». Поле «центр» в свою очередь состоит еще из двух полей: «координата X» и «координата Y». Ввести информацию по 10 окружностям. Вывести величину среднего радиуса всех окружностей .Далее вывести координаты центра окружностей чей радиус выше среднего.

11. Определить комбинированный (структурный) тип для представления анкеты жителя, состоящей из его фамилии, названия города, где он проживает, и городского адреса. Адрес состоит из полей: «улица», «дом», «квартира». Ввести информацию по 100 жителям. Вывести фамилии жителей, которые живут в Воронеже на улице Лизюкова.

12. Определить комбинированный (структурный) тип, описывающий окружность и состоящий из двух полей: «радиус» и «центр». Поле «центр» в свою очередь состоит еще из двух полей: «координата X» и «координата Y». Ввести информацию по 10 окружностям. Вывести радиус окружности, чей центр самый удаленный от оси OY(оси ординат).

13. Определить комбинированный (структурный) тип для представления студенческой ведомости состоящей из 2х полей: Ф. И. О., и оценка. В свою очередь поле оценка состоит из 4 элементов: оценка за математику, оценка за физику, оценка за информатику и средний балл. Составить программу, позволяющую вводить студенческую ведомость (без среднего балла). Далее найти для каждого студента средний балл. Вывести фамилии отличников по математике.

14. Определить комбинированный (структурный) тип, описывающий окружность и состоящий из двух полей: «радиус» и «центр». Поле «центр» в свою очередь состоит еще из двух полей: «координата X» и «координата Y». Ввести информацию по 10 окружностям. Вывести радиус окружности, чей центр расположен ближе всего к оси OY(оси ординат).

15. Определить комбинированный (структурный) тип для представления анкеты студента, состоящей из его фамилии, дня рождения и пола. «День рождения» состоит из полей: «число», «месяц», «год». Ввести информацию по 25 студентам из группы. Вывести все фамилии девочек, родившихся в марте 1991 года.

16. Определить комбинированный (структурный) тип для представления информации о кости домино, состоящей из левой половинки и правой половинки. Поля «левая» и «правая» половинки хранят информацию о количестве точек на половинках. Описать массив из 28 элементов (костей домино). Заполнить массив случайными числами или ввести с клавиатуры. Определить, правильно ли выставлены кости в данном массиве (Равна ли правая цифра очередной кости левой цифре следующей кости).

Задача №3

1. Определить комбинированный (структурный) тип для представления анкеты жителя, состоящей из его фамилии, названия города, где он проживает, и городского адреса. Адрес состоит из полей: «улица», «дом», «квартира». Ввести информацию по 10 жителям. Переписать из исходного массива в другой массив, информацию только о тех жителях, которые живут в Москве.
2. Определить комбинированный (структурный) тип, описывающий окружность и состоящий из двух полей: «радиус» и «центр». Поле «центр» в свою очередь состоит еще из двух полей: «координата X» и «координата Y». Ввести информацию по 10 окружностям. Переписать из исходного массива в другой массив, информацию только о тех окружностях, центр которых лежит в 1-ой четверти.
3. Определить комбинированный (структурный) тип для представления анкеты студента, состоящей из его фамилии, дня рождения и пола. «День рождения» состоит из полей: «число», «месяц», «год». Ввести информацию по 25 студентам из группы. Переписать из исходного массива в другой массив, информацию только о тех студентах, которые родились в указанном году. (указанную год вводит пользователь клавиатуры)
4. Определить комбинированный (структурный) тип для представления анкеты жителя, состоящей из его фамилии, названия города, где он проживает, и городского адреса. Адрес состоит из полей: «улица», «дом», «квартира». Ввести информацию по 10 жителям. Переписать из исходного массива в другой массив, информацию только о тех жителях, фамилия которых начинается на указанную букву (указанную букву вводит пользователь клавиатуры)
5. Определить комбинированный (структурный) тип для представления студенческой ведомости состоящей из 2х полей: Ф. И. О., и оценка. В свою очередь поле оценка состоит из 4 элементов: оценка за математику, оценка за физику, оценка за информатику и средний балл. Составить программу, позволяющую вводить студенческую ведомость (без среднего балла). Переписать из исходного массива в другой массив, информацию только о студентах-незадолжниках (у которых нет ни одной двойки),
6. Определить комбинированный (структурный) тип, описывающий окружность и состоящий из двух полей: «радиус» и «центр». Поле «центр» в свою очередь состоит еще из двух полей: «координата X» и «координата Y». Ввести информацию по 10 окружностям. Переписать из исходного массива в другой массив, информацию только о тех окружностях, радиус которых больше 1.
7. Определить комбинированный (структурный) тип для представления анкеты студента, состоящей из его фамилии, дня рождения и пола. «День рождения» состоит из полей: «число», «месяц», «год». Ввести информацию по 25 студентам из группы. Переписать из исходного массива в другой массив, информацию только о тех студентах, фамилия которых начинается на указанную букву (указанную букву вводит пользователь клавиатуры)

8. Определить комбинированный (структурный) тип для представления анкеты жителя, состоящей из его фамилии, названия города, где он проживает, и городского адреса. Адрес состоит из полей: «улица», «дом», «квартира». Ввести информацию по 100 жителям. Переписать из исходного массива в другой массив, информацию только о тех жителях, которые живут в Ростове-на-Дону на улице Ленина.

9. Определить комбинированный (структурный) тип, описывающий окружность и состоящий из двух полей: «радиус» и «центр». Поле «центр» в свою очередь состоит еще из двух полей: «координата X» и «координата Y». Ввести информацию по 10 окружностям. Переписать из исходного массива в другой массив, информацию только о тех окружностях, центр которых лежит в 3-ой четверти.

10. Определить комбинированный (структурный) тип для представления анкеты студента, состоящей из его фамилии, дня рождения и пола. «День рождения» состоит из полей: «число», «месяц», «год». Ввести информацию по 25 студентам из группы. Переписать из исходного массива в другой массив, информацию только о тех студентах, которые родились в мае 1986 года и являются мальчиками .

11. Определить комбинированный (структурный) тип, описывающий окружность и состоящий из двух полей: «радиус» и «центр». Поле «центр» в свою очередь состоит еще из двух полей: «координата X» и «координата Y». Ввести информацию по 10 окружностям. Вывести величину среднего радиуса всех окружностей . Переписать из исходного массива в другой массив, информацию только о тех окружностях, радиус которых меньше 1.

12. Определить комбинированный (структурный) тип для представления анкеты жителя, состоящей из его фамилии, названия города, где он проживает, и городского адреса. Адрес состоит из полей: «улица», «дом», «квартира». Ввести информацию по 100 жителям . Переписать из исходного массива в другой массив, информацию только о тех жителях, которые живут в Воронеже на улице Лизюкова.

13. Определить комбинированный (структурный) тип, описывающий окружность и состоящий из двух полей: «радиус» и «центр». Поле «центр» в свою очередь состоит еще из двух полей: «координата X» и «координата Y». Ввести информацию по 10 окружностям. . Переписать из исходного массива в другой массив, только информацию о тех окружностях, центр которых лежит выше оси OX.

14. Определить комбинированный (структурный) тип для представления анкеты студента, состоящей из его фамилии, дня рождения и пола. «День рождения» состоит из полей: «число», «месяц», «год». Ввести информацию по 25 студентам из группы. Переписать из исходного массива в другой массив, информацию только о тех студентах, которые родились в марте 1991 года и являются девочками

15. Определить комбинированный (структурный) тип, описывающий окружность и состоящий из двух полей: «радиус» и «центр». Поле «центр» в свою очередь состоит еще из двух полей: «координата X» и «координата Y». Ввести информацию по 10 окружностям. Переписать из исходного массива в другой массив, информацию только о тех окружностях, центр которых лежит ниже оси OX.

Задача №4

1) Определить структурный тип, описывающий гостиничный номер (название гостиницы, номер, комфортность (люкс, полулюкс стандарт, эконом), количество человек, стоимость). Заполнить структурный массив 10-ю записями. Переписать из

исходного массива в другой массив, информацию только о тех гостиничных номерах, название гостиницы которых начинается с сочетания букв «City». Затем новый массив отсортировать по номеру. (рационально переставлять все поля структуры разом)

2) Определить структурный тип, описывающий расписание полетов самолетов (пункт назначения, время отправления, время прибытия, время полета, стоимость билета). Заполнить структурный массив 10-ю записями. Переписать из исходного массива в другой массив, информацию только о тех рейсах пункт назначения, которых содержит по 2 буквы «а». Затем новый массив отсортировать по пункту назначения по алфавиту. (рационально переставлять все поля структуры разом)

3) Определить структурный тип, описывающий гостиничный номер (название гостиницы, номер, комфортность (люкс, полулюкс стандарт, эконом), количество человек, стоимость). Заполнить структурный массив 10-ю записями. Переписать из исходного массива в другой массив, информацию только о тех гостиничных номерах, в название гостиницы которых есть по 2 буквы «а». Затем новый массив отсортировать по названию гостиницы по алфавиту. (рационально переставлять все поля структуры разом)

4) Определить структурный тип, описывающий музыкальные CD-диски (название альбома, исполнитель, стиль, год выпуска, длительность, стоимость). Заполнить структурный массив 10-ю записями. Переписать из исходного массива в другой массив, информацию только о тех CD-дисках, название альбома которых начинается на сочетание букв (3 - 4) введенных пользователем. Затем новый массив отсортировать по стилю по алфавиту. (рационально переставлять все поля структуры разом)

5) Определить структурный тип, описывающий книги домашней библиотеки (автор, название книги, издательство, год издания, стоимость). Заполнить структурный массив 10-ю записями. Переписать из исходного массива в другой массив, информацию только о тех книгах, в название которых есть по 3 буквы «о». Затем вывести информацию, отсортированную по названию издательства по алфавиту. (рационально переставлять все поля структуры разом)

6) Определить структурный тип, описывающий книги домашней библиотеки (автор, название книги, издательство, год издания, стоимость). Заполнить структурный массив 10-ю записями. Переписать из исходного массива в другой массив, информацию только о тех книгах, в название издательства которых есть 1 буква «к». Затем новый массив отсортировать по названию книги по алфавиту. (рационально переставлять все поля структуры разом)

7) Определить структурный тип, описывающий гостиничный номер (название гостиницы, номер, комфортность (люкс, полулюкс стандарт, эконом), количество человек, стоимость). Заполнить структурный массив 10-ю записями. Переписать из исходного массива в другой массив, информацию только о тех гостиничных номерах, название гостиницы которых начинается на букву «Р». Затем новый массив отсортировать по возрастанию стоимости. (рационально переставлять все поля структуры разом)

8) Определить структурный тип, описывающий книги домашней библиотеки (автор, название книги, издательство, год издания, стоимость). Заполнить структурный массив 10-ю записями. Переписать из исходного массива в другой массив, только информацию только о тех книгах, название которых начинается на «Фент». Затем новый массив отсортировать по стоимости. (рационально переставлять все поля структуры разом)

9) Определить структурный тип, описывающий гостиничный номер (название гостиницы, номер, комфортность (люкс, полулюкс стандарт, эконом), количество человек, стоимость). Заполнить структурный массив 10-ю записями. Переписать из исходного массива в другой массив, информацию только о тех гостиничных номерах, название гостиницы которых оканчивается на сочетание букв «plaza». Затем новый массив отсортировать по возрастанию стоимости. (рационально переставлять все поля структуры разом)

10) Определить структурный тип, описывающий музыкальные CD-диски (название альбома, исполнитель, стиль, год выпуска, длительность, стоимость). Заполнить структурный массив 10-ю записями. Переписать из исходного массива в другой массив, информацию только о тех CD-дисках, название стиля которых начинается на сочетание букв (3 - 4) введенных пользователем. Затем новый массив отсортировать по исполнителю по алфавиту. (рационально переставлять все поля структуры разом)

11) Определить структурный тип, описывающий гостиничный номер (название гостиницы, номер, Комфортность (люкс, полулюкс стандарт, эконом), количество человек, стоимость). Заполнить структурный массив 10-ю записями. Переписать из исходного массива в другой массив, информацию только о тех гостиничных номерах, название гостиницы которых оканчивается на сочетание букв «hostel». Затем новый массив отсортировать по комфортности по алфавиту. (рационально переставлять все поля структуры разом)

12) Определить структурный тип, описывающий книги домашней библиотеки (автор, название книги, издательство, год издания, стоимость). Заполнить структурный массив 10-ю записями. Переписать из исходного массива в другой массив, только информацию только о тех книгах, название издательства которых начинается на «Ф». Затем новый массив отсортировать по названию книги по алфавиту. (рационально переставлять все поля структуры разом)

13) Определить структурный тип, описывающий музыкальные CD-диски (название альбома, исполнитель, стиль, год выпуска, длительность, стоимость). Заполнить структурный массив 10-ю записями. Переписать из исходного массива в другой массив, информацию только о тех CD-дисках исполнитель, которых начинается на букву «Б». Затем новый массив отсортировать по стоимости. (рационально переставлять все поля структуры разом)

14) Определить структурный тип, описывающий студенческую ведомость (Ф. И. О., оценки за три экзамена, средний балл). Заполнить структурный массив 10-ю записями. Переписать из исходного массива в другой массив, информацию только о тех студентах фамилии, которых оканчиваются на «ова». Затем новый массив отсортировать по среднему баллу. (рационально переставлять все поля структуры разом)

15) Определить структурный тип, описывающий расписание полетов самолетов (пункт посадки, время отправления, время прибытия, время полета, стоимость билета). Заполнить структурный массив 10-ю записями. Переписать из исходного массива в другой массив, информацию только о тех рейсах пункт назначения, которых оканчивается сочетанием «град». Затем новый массив отсортировать по времени полета. (рационально переставлять все поля структуры разом)

Задача №5

1) Определить комбинированный (структурный) тип для представления информации по горным вершинам, состоящей из названия вершины и ее высоты. Ввести информацию по 20 вершинам. Вывести среднее значение высот всех 20 вершин. Затем вывести информацию, отсортированную по возрастанию высоты вершины. (рационально переставлять все поля структуры разом)

2) Определить комбинированный (структурный) тип, описывающий окружность и состоящий из двух полей: «радиус» и «центр». Поле «центр» в свою очередь состоит еще из двух полей: «координата X» и «координата Y». Ввести информацию по 10 окружностям. Вывести радиус самой большой окружности. Затем вывести информацию, отсортированную по возрастанию радиуса окружности. (рационально переставлять все поля структуры разом)

3) Определить комбинированный (структурный) тип для представления информации по горным вершинам, состоящей из названия вершины и ее высоты. Ввести информацию по 10 вершинам. Вывести название самой высокой вершины из всех 10. Затем вывести информацию, отсортированную по возрастанию высоты вершины. (рационально переставлять все поля структуры разом)

4) Определить комбинированный (структурный) тип, описывающий окружность и состоящий из двух полей: «радиус» и «центр». Поле «центр» в свою очередь состоит еще из двух полей: «координата X» и «координата Y». Ввести информацию по 10 окружностям. Вывести радиус окружности, чей центр расположен ближе всего к оси OX(оси абсцисс). Затем вывести информацию, отсортированную по возрастанию радиуса окружности. (рационально переставлять все поля структуры разом)

5) Определить комбинированный (структурный) тип для представления информации по горным вершинам, состоящей из названия вершины и ее высоты. Ввести информацию по 50 вершинам. Вывести название самой низкой вершины из всех 50. Затем вывести информацию, отсортированную по возрастанию высоты вершины. (рационально переставлять все поля структуры разом)

6) Определить комбинированный (структурный) тип, описывающий окружность и состоящий из двух полей: «радиус» и «центр». Поле «центр» в свою очередь состоит еще из двух полей: «координата X» и «координата Y». Ввести информацию по 10 окружностям. Вывести сумму радиусов всех окружностей. Затем вывести информацию, отсортированную по возрастанию радиуса окружности. (рационально переставлять все поля структуры разом)

7) Определить комбинированный (структурный) тип, описывающий окружность и состоящий из двух полей: «радиус» и «центр». Поле «центр» в свою очередь состоит еще из двух полей: «координата X» и «координата Y». Ввести информацию по 10 окружностям. Вывести радиус окружности, чей центр самый удаленный от оси OY(оси ординат). Затем вывести информацию, отсортированную по возрастанию радиуса окружности. (рационально переставлять все поля структуры разом)

8) Определить комбинированный (структурный) тип, описывающий окружность и состоящий из двух полей: «радиус» и «центр». Поле «центр» в свою очередь состоит еще из двух полей: «координата X» и «координата Y». Ввести информацию по 10 окружностям. Вывести радиус окружности, чей центр расположен ближе всего к оси OY(оси ординат). Затем вывести информацию, отсортированную по возрастанию радиуса окружности. (рационально переставлять все поля структуры разом)

9) Определить комбинированный (структурный) тип для представления анкеты ребенка, состоящей из его имени, пола и роста. Ввести информацию по 20 детям. Вывести средний рост девочек . Затем вывести информацию, отсортированную по имени по алфавиту. (рационально переставлять все поля структуры разом)

10) Определить комбинированный (структурный) тип, описывающий окружность и состоящий из двух полей: «радиус» и «центр». Поле «центр» в свою очередь состоит еще из двух полей: «координата X» и «координата Y». Ввести информацию по 10 окружностям. Вывести радиус самой маленькой окружности. Затем вывести информацию, отсортированную по возрастанию радиуса окружности. (рационально переставлять все поля структуры разом)

11) .Определить комбинированный (структурный) тип, описывающий окружность и состоящий из двух полей: «радиус» и «центр». Поле «центр» в свою очередь состоит еще из двух полей: «координата X» и «координата Y». Ввести информацию по 10 окружностям. Вывести радиус окружности, чей центр самый удаленный от начала координат. Затем вывести информацию, отсортированную по возрастанию радиуса окружности. (рационально переставлять все поля структуры разом)

12) Определить комбинированный (структурный) тип для представления анкеты ребенка, состоящей из его имени, пола и роста. Ввести информацию по 20 детям. Вывести имя самого высокого мальчика. Затем вывести информацию, отсортированную по имени по алфавиту. (рационально переставлять все поля структуры разом)

13) Определить комбинированный (структурный) тип, описывающий окружность и состоящий из двух полей: «радиус» и «центр». Поле «центр» в свою очередь состоит еще из двух полей: «координата X» и «координата Y». Ввести информацию по 10 окружностям. Вывести величину среднего радиуса всех окружностей. Затем вывести информацию, отсортированную по возрастанию радиуса окружности. (рационально переставлять все поля структуры разом)

14) Определить комбинированный (структурный) тип, описывающий окружность и состоящий из двух полей: «радиус» и «центр». Поле «центр» в свою очередь состоит еще из двух полей: «координата X» и «координата Y». Ввести информацию по 10 окружностям. Вывести радиус окружности, чей центр самый удаленный от оси OX(оси абсцисс). (рационально переставлять все поля структуры разом)

ЛАБОРАТОРНАЯ РАБОТА №7. ФУНКЦИИ, ОПРЕДЕЛЕННЫЕ ПОЛЬЗОВАТЕЛЕМ

Цель лабораторной работы

Разработка программ языке С++ с использованием функций, определенных пользователем.

Методические указания

ФУНКЦИИ, ОПРЕДЕЛЯЕМЫЕ ПОЛЬЗОВАТЕЛЕМ

Сложная задача может быть разделена на более простые и обозримые с помощью функций, что ведет к упрощению её структуры и позволяет избежать избыточности кода за счет многократно вызываемых функций из разных точек программы.

Функция – это именованная последовательность описаний и операторов, выполняющих какое-либо законченное действие. Функция может принимать *параметры* и *возвращать вычисленное значение* (результат функции), а также изменять параметры, если они передаются по ссылке или через указатель.

Любая программа на языке С/С++ состоит из функций, одна из которых должна иметь имя **main** (с неё начинается выполнение программы). Пользовательскую функцию можно определить либо в том же файле, где находится главная функция main( ), либо в отдельном файле .

ФУНКЦИИ, ВОЗВРАЩАЮЩИЕ ВЫЧИСЛЕННОЕ ЗНАЧЕНИЕ

Формат объявления функции:

```
тип имя (список_параметров_и_указание_их_типов)
{
        тело функции
        return (возвращаемое_ значение);
}
```

В заголовке функции указывается, во-первых, тип возвращаемого ею зна-чения. Во-вторых, указывается имя функции, по которому будет происходить обращение к ней из другой функции. В-третьих, список принимаемых параметров или аргументов и указание их типов. Данные параметры называются формальными параметрами. Далее располагается тело функции, имеющее такую же структуру, что и функция **main()**. Оно может содержать описание локальных переменных, также исполняемые операторы и оператор **return (),** в котором определяется возвращаемое функцией значение.

Вызов функции происходит в операторе присвоения или может входить в состав выражений. После имени функции в скобках указываются фактические параметры функции, разделенные запятыми. При вызове функции типы и порядок следования параметров должен совпадать с объявленным.

Пример: Вычислить величину *Q*, используя функции для выполнения схожих действий.

$$Q = \begin{cases} \dfrac{\min(a,b)+1}{\min(\sin a, \sin b) - c^2}, & если \quad c > \min(a,b); \\ \dfrac{e^{d+1} + \min^3(c,d)}{\cos^3 a + \min(\sqrt{a}, \sqrt{b})}, & если \ a < \min(c,d); \end{cases}$$

```
#include<stdoi.h>
#include<math.h>
float min ( float x , float y) /*заголовок функции; x и y – формальные параметры */
{float m; //объявление локальной переменной
if (x > y) m = y; else m = x; //вычисление минимального значения
return ( m );
}// возвращаемое значение

void main( ) //главная программа-функция
{ float z, a, b, c, d, Q;
printf (“wwedite a, b, c, d,”);
scanf ( “ %f %f ”, &a, &b );
 z = min(a, b); /* обращение к функции min, где a и b – фактические параметры в отличии от x и y – формальных.*/
if (c > z) Q = (z+1)(/min(sin(a),sin(b)) - c*c );
if (a < min (c, d)) Q = (exp(d+1)+ pow( min(c,d), 3)) / (pow(cos(a),3) + min(sqrt(a),sqrt(b)));
printf (“ Q = %f ”,Q);
}
```

При вызове подпрограммы функции управление передается из главной программы в подпрограмму, при этом вместо формальных параметров **x** и **y** подставляются фактические параметры **a** и **b** со своими значениями, далее производится вычисление согласно операторам, записанным в тексте функции. Далее с помощью оператора **return** возвращается некое полученное значение **m** в главную программу **main** в точку выхода.

Варианты заданий

Задание №1

1. Создать функцию, которая возвращает меньшее из двух данных чисел. Для создаваемой функции: подобрать имя; указать тип функции; выбрать имена и типы входных параметров; описать тело функции с обязательным оператором в конце; в главной программе вызвать созданную функцию два раза с различными входными данными. Вывести результаты в главной программе.
2. Создать функцию, которая переводит время, заданное в минутах в секунды. Для создаваемой функции: подобрать имя; указать тип функции; выбрать имена и типы входных параметров; описать тело функции с обязательным оператором в конце; в главной программе вызвать созданную функцию два раза с различными входными данными. Вывести результаты в главной программе.
3. Создать функцию, которая определяет периметр треугольника по трем его сторонам. Для создаваемой функции: подобрать имя; указать тип функции; выбрать имена и типы входных параметров; описать тело функции с обязательным оператором в конце; в главной программе вызвать созданную функцию два раза с различными входными данными. Вывести результаты в главной программе.
4. Создать функцию, которая возвращает номер квадранта, в котором находится точка. Для создаваемой функции: подобрать имя; указать тип функции; выбрать имена и типы входных параметров; описать тело функции с обязательным оператором в конце; в главной программе вызвать созданную функцию два раза с различными входными данными. Вывести результаты в главной программе.

5. Создать функцию, которая возвращает среднее арифметическое трех данных чисел. Для создаваемой функции: подобрать имя; указать тип функции; выбрать имена и типы входных параметров; описать тело функции с обязательным оператором в конце; в главной программе вызвать созданную функцию два раза с различными входными данными. Вывести результаты в главной программе.

6. Создать функцию, которая определяет площадь круга по его радиусу. Для создаваемой функции: подобрать имя; указать тип функции; выбрать имена и типы входных параметров; описать тело функции с обязательным оператором в конце; в главной программе вызвать созданную функцию два раза с различными входными данными. Вывести результаты в главной программе.

7. Создать функцию, которая возвращает остаток от деления двух натуральных чисел Для создаваемой функции: подобрать имя; указать тип функции; выбрать имена и типы входных параметров; описать тело функции с обязательным оператором в конце; в главной программе вызвать созданную функцию два раза с различными входными данными. Вывести результаты в главной программе.

8. Создать функцию, которая переводит радианы в градусы. Для создаваемой функции: подобрать имя; указать тип функции; выбрать имена и типы входных параметров; описать тело функции с обязательным оператором в конце; в главной программе вызвать созданную функцию два раза с различными входными данными. Вывести результаты в главной программе.

9. Создать функцию, которая определяет длину отрезка по его координатам. Для создаваемой функции: подобрать имя; указать тип функции; выбрать имена и типы входных параметров; описать тело функции с обязательным оператором в конце; в главной программе вызвать созданную функцию два раза с различными входными данными. Вывести результаты в главной программе.

10. Создать функцию, которая возвращает в долларах сумму, заданную в рублях. Для создаваемой функции: подобрать имя; указать тип функции; выбрать имена и типы входных параметров; описать тело функции с обязательным оператором в конце; в главной программе вызвать созданную функцию два раза с различными входными данными. Вывести результаты в главной программе.

11. Создать функцию, которая возвращает большее из двух данных чисел. Для создаваемой функции: подобрать имя; указать тип функции; выбрать имена и типы входных параметров; описать тело функции с обязательным оператором в конце; в главной программе вызвать созданную функцию два раза с различными входными данными. Вывести результаты в главной программе.

12. Создать функцию, которая определяет длину окружности по заданному радиусу. Для создаваемой функции: подобрать имя; указать тип функции; выбрать имена и типы входных параметров; описать тело функции с обязательным оператором в конце; в главной программе вызвать созданную функцию два раза с различными входными данными. Вывести результаты в главной программе.

13. Создать функцию, которая переводит скорость из км/час в м/сек. Для создаваемой функции: подобрать имя; указать тип функции; выбрать имена и типы входных параметров; описать тело функции с обязательным оператором в конце; в главной программе вызвать созданную функцию два раза с различными входными данными. Вывести результаты в главной программе.

14. Создать функцию, которая возвращает среднее геометрическое двух данных чисел. Для создаваемой функции: подобрать имя; указать тип функции; выбрать имена и типы входных параметров; описать тело функции с обязательным оператором в

конце; в главной программе вызвать созданную функцию два раза с различными входными данными. Вывести результаты в главной программе.

15. Создать функцию, которая возвращает в рублях сумму, заданную в долларах. Для создаваемой функции: подобрать имя; указать тип функции; выбрать имена и типы входных параметров; описать тело функции с обязательным оператором в конце; в главной программе вызвать созданную функцию два раза с различными входными данными. Вывести результаты в главной программе.

16. Создать функцию, которая переводит градусы в радианы. Для создаваемой функции: подобрать имя; указать тип функции; выбрать имена и типы входных параметров; описать тело функции с обязательным оператором в конце; в главной программе вызвать созданную функцию два раза с различными входными данными. Вывести результаты в главной программе.

17. Создать функцию, которая переводит время, заданное в секундах в минуты. Для создаваемой функции: подобрать имя; указать тип функции; выбрать имена и типы входных параметров; описать тело функции с обязательным оператором в конце; в главной программе вызвать созданную функцию два раза с различными входными данными. Вывести результаты в главной программе.

Задание №2.

Вариант№ 1 Реализовать функцию. Функция вычисляет площадь параллелограмма s= a b Cos α по заданным двум сторонам и углу между ними. В главной программе задано два параллелограмма. Найти их площади, вызвав функцию 2 раза.

Вариант №2 Реализовать функцию . Функция вычисляет площадь квадрата по заданной стороне. В главной программе задано два квадрата. Найти их площади , вызвав функцию 2 раза.

Вариант №3 Реализовать функцию . Функция вычисляет диагональ прямоугольника по заданным двум сторонам . В главной программе задано два прямоугольника . Найти их диагонали, вызвав функцию 2 раза.

Вариант№ 4 Реализовать функцию . Функция вычисляет площадь s= ½ (a +b) h.. равнобедренной трапеции по заданным двум основаниям a,b и высоте h. ..В главной программе задано две равнобедренные трапеции . Найти их площади, вызвав функцию 2 раза.

Вариант№ 5. Реализовать функцию. Функция вычисляет площадь окружности по заданному радиусу R. В главной программе задано две окружности . Найти их площади , вызвав функцию 2 раза.

Вариант№ 6. Реализовать функцию . Функция вычисляет объем сферы $v=4/3\ \pi\ R^3$ по заданному радиусу R. В главной программе задано две сферы. Найти их объемы, вызвав функцию 2 раза.

Вариант№ 7. Реализовать функцию . Функция вычисляет объем квадратной призмы по заданной высоте H и стороне основания a. В главной программе задано две квадратные призмы . Найти их объемы , вызвав функцию 2 раза.

Вариант№ 8. Реализовать функцию . Функция вычисляет объем правильной треугольной призмы по заданной высоте H и стороне основания a . .В главной программе задано две треугольные призмы . Найти их объемы , вызвав функцию 2 раза.

Вариант№9 Реализовать функцию . Функция вычисляет объем цилиндра по заданной высоте H и радиусу основания R . В главной программе задано два цилиндра . Найти их объемы , вызвав функцию 2 раза.

Вариант№11 Реализовать функцию . Функция вычисляет площадь боковой поверхности S=π R $\sqrt{R^2+H^2}$ конуса по заданной высоте H и радиусу основания R . В главной программе задано два конуса . Найти их площади боковых поверхностей, вызвав функцию 2 раза.

Вариант№11 Реализовать функцию . Функция вычисляет объем v= 1/3 H π R^2 конуса по заданной высоте H и радиусу основания R . В главной программе задано два конуса . Найти их объемы, вызвав функцию 2 раза.

Вариант№13. Реализовать функцию . Функция вычисляет объем v=1/3 H π (R^2 +r^2+R r) Усеченного конуса по заданной высоте H и двум радиусам оснований R и r . .В главной программе задано два конуса . Найти их объемы , вызвав функцию 2 раза.

Вариант№14 Реализовать функцию . Функция вычисляет длину вектора на плоскости по заданным координатам x и y . В главной программе задано два вектора . Найти их длины, вызвав функцию 2 раза.

Вариант№15 Реализовать функцию . Функция вычисляет тангенс угла наклона (между осью OX и вектором) вектора на плоскости по заданным координатам x и y . В главной программе задано два вектора . Найти их углы, вызвав функцию 2 раза.

Вариант№16. Реализовать функцию . Функция вычисляет расстояние до начала координат от точки на плоскости по заданным её координатам x и y . В главной программе задано две точки . Найти их расстояния до начала координат , вызвав функцию 2 раза.

Задача 3

1. Описать функцию Sign(X) целого типа, возвращающую для вещественного числа X его знак , то есть следующие значения: -1, если X<0, 0, если X=0; 1, если X>0. С помощью этой функции найти значение выражения Sign(A) + Sign(B) для данных вещественных чисел A и B.

2. Описать функцию RootsCount(A, B, C) целого типа, определяющую количество корней квадратного уравнения $A*x^2 + B*x + C = O$ (A, B, C — вещественные параметры, A <> 0). С ее помощью найти количество корней для каждого из трех квадратных уравнений с данными коэффициентами. Количество корней определять по значению дискриминанта: $D = B^2 – 4*A*C$.

3. Описать функцию CircleS(R) вещественного типа, находящую площадь круга радиуса R (R — вещественное). С помощью этой функции найти площади трех кругов с данными радиусами. Площадь круга радиуса R вычисляется по формуле $S = p*R^2$. В качестве значения p использовать 3,14. ,

4. Описать функцию RingS(R1,R2) вещественного типа, находящую площадь кольца, заключенного между двумя окружностями с общим центром и радиусами R1 и R2 (R1 и R2 — вещественные, R1 > R2). С ее помощью найти площади трех колец, для которых даны внешние и внутренние радиусы. Воспользоваться формулой площади круга радиуса R: $S = p*R^2$. В качестве значения p использовать 3.14.

5. Описать функцию TriangleP(a, h), находящую периметр равнобедренного треугольника по его основанию a и высоте h, проведенной к основанию (a и h — вещественные). С помощью этой, функции найти периметры трех треугольников, для которых даны основания и высоты. Для нахождения боковой стороны b треугольника использовать теорему Пифагора: $b^2 = (a/2)^2 + h^2$.

6. Описать функцию Calc(A,B,Op) вещественного типа, выполняющую над ненулевыми вещественными числами A и B одну из арифметических операций и возвращающую ее результат. Вид операции определяется целым параметром Op: 1 — вычитание, 2 — умножение, 3 — деление, остальные значения — сложение. С помощью Calc выполнить для данных A и B операции, определяемые данными целыми N1,N2,N3.

7. Описать функцию Quarter (x, y) целого типа, определяющую номер координатной четверти, в которой находится точка с ненулевыми вещественными координатами (x, y). С помощью этой функции найти номера координатных четвертей для трех точек с данными ненулевыми координатами.

8. Вычисление скалярного произведения векторов (массивов) оформить в виде функции (x,y)- скалярное произведение векторов – это сумма поэлементных произведений их компонент. В главной программе заданы два вектора (массива) $x = (x_1, x_2, x_3, x_4)$, и $y = (y_1, y_2, y_3, y_4)$. Определить косинус угла α между векторами x и y по формуле: $\cos\alpha = \frac{(x,y)}{\sqrt{(x,x)(y,y)}}$, где $(x,y) = x_1y_1+x_2y_2+\ldots x_ny_n$ - скалярное произведение векторов (сумма поэлементных произведений компонент), используя описанную функцию 3 раза.

9. Даны длины сторон треугольника a, b, c. Найти медианы треугольника, сторонами которого являются медианы исходного треугольника. Для вычисления медианы проведенной к стороне a, использовать формулу $0{,}5\sqrt{2b^2+2c^2-a^2}$ Вычисление медианы оформить в виде функции.

10. Описать функцию DegToRad(D) вещественного типа, находящую величину угла в радианах, если дана его величина D в градусах (D — вещественное число, O < D < 360). Воспользоваться следующим соотношением: 180° = p радианов. В качестве значения p использовать 3.14. С помощью функции DegToRad перевести из градусов в радианы пять данных углов.

11. Описать функцию RadToDeg(R) вещественного типа, находящую величину угла в градусах, если дана его величина R в радианах (R — вещественное число, O < R < 2p). Воспользоваться следующим соотношением: 180° = p радианов. В качестве значения p использовать 3.14. С помощью функции RadToDeg перевести из радианов в градусы пять данных углов.

12. Четыре точки заданы своими координатами X(x1, x2), Y(y1, y2), Z(z1, z2), P(p1, p2). Выяснить, какие из них находятся на максимальном расстоянии друг от друга и вывести на печать значение этого расстояния. Вычисление расстояния между двумя точками оформить в виде функции.

13. Четыре точки заданы своими координатами X(x1, x2, x3), Y(y1, y2, y3), Z(z1, z2, z3), T(t1, t2, t3).Выяснить, какие из них находятся на минимальном расстоянии друг от друга и вывести на печать значение этого расстояния. Вычисление расстояния между двумя точками оформить в виде функции.

14. Описать функцию Power2(A, N) вещественного типа, находящую величину A^N (A — вещественный, N — целый параметр) по следующим формулам: $A^\circ=1$
A^N = A*A*...*A (N сомножителей), если N>0;
A^N= 1/(A*A*...*A) (|N| сомножителей), если N<0.
С помощью этой функции найти A^K, A^L, A^M, если даны числа A, K, L, M.

Задача № 4

1. Даны действительные числа a, b, c. Получить: $\frac{\max(a, a+b)+\max(a, b+c)}{1+\max(a+b\cdot c, b\cdot 15)}$

Описать функции нахождения наибольшего числа из 2 заданных величин.

2. Описать функцию Power1(A,B) вещественного типа, находящую величину A^B по формуле A^B - exp(B*ln(A)) (параметры A и B — вещественные). В случае нулевого или отрицательного параметра A функция должна возвращать 0. С помощью этой функции в главной программе найти степени A^P, B^P,C^P , если даны числа P, A, B, C.

3. Даны действительные числа a, b. Получить u = min(a, b-a), y = min(ab, a+b), k = min(u+y^2, 3.14). Описать функции нахождения минимального числа из 2 заданных величин.

4. Даны действительные числа s, t. Получить: g(1.2, s)+g(t, s)-g(2s-1, 5t), где $g(a,b)=\frac{a^2+b^2}{a^2+2ab+3b^2+4}$ - функция

5. Даны действительные числа x, y. Получить: f(x, -2y, 1.17)+f(2.2, x, x-y), где $f(a,b,c)=\frac{2a-b-\sin c}{5+|c|}$. - функция

6. Даны натуральные числа a, b, c. Найти НОД(a, b, c), используя формулу: НОД(a, b, c) = НОД((a, b), c). Алгоритм Евклида: НОД(A, B) = НОД(B, A mod B), если B <> 0; НОД(A, 0) = A. Описать функцию НОД(A, B), используя цикл while.

7. Получить для x = 1, 3, 4, 5 все значения выражения Q(x)=p(x+1) - p(x) вывести на экран, где p(y) = a3y^3 + a2y^2 + a1y + a0 ;- функция (где a3 , a2 , a1 , a0. определенные константы)

8. Даны действительные числа x, y. Получить: Q=tg(f(x+y, xy, y-x)+f(3.1, 1.4, y-sinx)), где $f(a,b,c)=\frac{3b+c-e^{-a}}{1+|\cos 3a|}$ - функция

9. Даны действительные числа a, b. Получить u = min(a, b*a), y = min(a-b, a+b), k = min(u^3+y^3, 28). Описать функции нахождения минимального числа из 2 заданных величин.

10. Даны действительные числа a, b. Получить r = max (a, b + a), d = max (ab, a + b), s = max (r + d^2, 3.14). Описать функции нахождения наибольшего числа из 2 заданных величин.

11. Даны действительные числа a_0, a_1, a_2, a_3. Получить для x = 2, 4, 7 значения Q(x)= p(x+1) + p(x), где p(y) = a_3y^2 + a_2y + 2(a_1 + a_0). - функция

12. Даны действительные числа s, t. Получить: Q= |g(ln(s, t+1))-g(t, s)|, где $g(a,b)=\frac{a^2+b^2}{a^2+2ab+3b^2+4}$ - функция.

13. Даны натуральные числа a, b, c. Найти НОД(a, b, c), используя формулу:.НОД(a, b, c) = НОД((a, b), c). Алгоритм Евклида: НОД(A, B) = НОД(B, A mod B), если B <> 0; НОД(A, 0) = A Описать функцию НОД(A, B), используя цикл while.

14. Три точки заданы своими координатами X(x1, x2), Y(y1, y2) и Z(z1, z2). Найти и напечатать координаты точки, для которой угол между осью абсцисс и лучом,

соединяющим начало координат с точкой, минимальный. Вычисление угла оформить в виде функции по формуле $\alpha = arctg\left(\frac{y}{x}\right)$. Вызвать функцию в программе 3 раза.

ЛАБОРАТОРНАЯ РАБОТА №8. ФУНКЦИИ И МАССИВЫ

Цель лабораторной работы

Разработка программ языке С++ с функциями, обрабатывающие массивы.

Методические указания

При передаче массивов в функции в качестве параметров передается в функцию не весь массив, а только *указатель на его начальный элемент.* В этом случае значения массива могут быть изменены внутри функции и возвращены в главную программу.

Пример:

Дано 3 одномерных массива *a, b, d* длиной 20 элементов каждый. Описать подходящую функцию для облегчения решения задачи. Вычислить

$$V = \begin{cases} \prod_{i=1}^{20} a_i - \prod_{i=1}^{20} (b_i + d_i), \text{если } \prod_{i=1}^{20} a_i > \prod_{i=1}^{20} d_i; \\ \prod_{i=1}^{20} (b_i - a_i) + \prod_{i=1}^{20} d_i, \text{иначе;} \end{cases}$$

```
#include<stdio.h>
 #include<conio.h>
 #include<math.h>
 #include<iostream.h>
 #include<time.h>
 #include<stdlib.h>
 float proizved (float *x, int n)
{
int i, float P = 1.0;
for (i = 0;i < n; i ++)
        P* = x[i];
return (P);
}
 void main()
 {
 float a[20],b[20],d[20],c[20],r[20],V;
 for (int i=0;i<20;i++)
 {      a[i] = rand()%10+5;
        b[i] = rand()%10+10;
        d[i] = rand()%5+15;
        c[i] = b[i]+d[i]; r[i] = b[i] - a[i];
 }
 if (proizved(a,20) > proizved(d, 20) )
        V = proizved(a, 20) – proizved(c, 20);
 else
        V = proizved (r, 20) + proizved (d, 20);
 cout << "V=" << V;
 getch();
 }
```

Объявление (прототип) функции должно быть расположено в тексте раньше её вызова, а определение функции может располагаться в любом месте текста программы или в другом файле.

Варианты заданий

Задача 1.

1. Оформить функцию поиска количества отрицательных элементов массива. В главной программе дано 3 одномерных массива a,b,c длиной 10 элементов каждый. Применить функцию для каждого из 3-х заданных массивов. (в функции не должно быть операторов ввода или вывода)
2. Оформить функцию поиска количества положительных элементов массива. В главной программе дано 2 одномерных массива x,y длиной 20 элементов каждый и один массив z длиной 5 элементов.. Применить функцию для каждого из 3-х заданных массивов. (в функции не должно быть операторов ввода или вывода)
3. Оформить функцию поиска количества элементов равных 5 . В главной программе дано 3 одномерных массива p,q,r длиной 10 элементов каждый. Применить функцию для каждого из 3-х заданных массивов. (в функции не должно быть операторов ввода или вывода)
4. Оформить функцию поиска количества нулевых элементов массива. В главной программе дано 3 одномерных массива arr1,arr2,arr3 длиной 10 элементов каждый. Применить функцию для каждого из 3-х заданных массивов. (в функции не должно быть операторов ввода или вывода)
5. Оформить функцию поиска количества элементов массива, больших заданного числа alfa. В главной программе дано 2 одномерных массива a,b длиной 15 элементов каждый. Применить функцию для каждого из 2-х заданных массивов. (в функции не должно быть операторов ввода или вывода)
6. Оформить функцию поиска суммы отрицательных элементов массива. В главной программе дано 2 одномерных массива x, y длиной 10 элементов каждый и один массив z длиной 5 элементов.. Применить функцию для каждого из 3-х заданных массивов. (в функции не должно быть операторов ввода или вывода)
7. Оформить функцию поиска суммы элементов массива больших 5. В главной программе дано 2 одномерных массива a,b,c длиной 10 элементов каждый. Применить функцию для каждого из 3-х заданных массивов. (в функции не должно быть операторов ввода или вывода)
8. Оформить функцию поиска суммы элементов массива больших 1. В главной программе дано 2 одномерных массива mass1, mass2 длиной 10 элементов каждый и один массив z длиной 5 элементов. Применить функцию для каждого из 3-х заданных массивов. (в функции не должно быть операторов ввода или вывода)
9. Оформить функцию поиска суммы элементов массива, больших заданного числа alfa. В главной программе дано 4 одномерных массива a,b,c,d длиной 10 элементов каждый. Применить функцию для каждого из 4-х заданных массивов. (в функции не должно быть операторов ввода или вывода)
10. Оформить функцию поиска суммы элементов массива, больших заданного числа alfa и меньшего заданного числа betta. В главной программе дано 3 одномерных массива a,b,c длиной 10 элементов каждый. Применить функцию для каждого из 3-х заданных массивов. (в функции не должно быть операторов ввода или вывода)

11. Оформить функцию поиска произведения положительных элементов массива. В главной программе дано 4 одномерных массива a,b,c,d длиной 10 элементов каждый. Применить функцию для каждого из4-х заданных массивов. (в функции не должно быть операторов ввода или вывода)

12. Оформить функцию поиска произведения отрицательных элементов массива. В главной программе дано 4 одномерных массива x,y,z,f длиной 10 элементов каждый. Применить функцию для каждого из 4-х заданных массивов. (в функции не должно быть операторов ввода или вывода)

13. Оформить функцию поиска произведения элементов массива, больших заданного числа alfa и меньшего заданного числа betta. В главной программе дано 2 одномерных массива a,b длиной 10 элементов каждый и один массив z длиной 5 элементов. Применить функцию для каждого из 3-х заданных массивов. (в функции не должно быть операторов ввода или вывода)

14. Оформить функцию поиска среднего арифметического отрицательных элементов массива. В главной программе дано 3 одномерных массива a,b,c длиной 10 элементов каждый. Применить функцию для каждого из 3-х заданных массивов. (в функции не должно быть операторов ввода или вывода)

15. Оформить функцию поиска среднего арифметического положительных элементов массива. В главной программе дано 3 одномерных массива a,b,c длиной 10 элементов каждый. Применить функцию для каждого из 3-х заданных массивов. (в функции не должно быть операторов ввода или вывода)

16. Оформить функцию поиска среднего геометрического элементов массива. В главной программе дано 3 одномерных массива a,b,c длиной 10 элементов каждый. Применить функцию для каждого из 3-х заданных массивов. (в функции не должно быть операторов ввода или вывода)

17. Оформить функцию поиска арифметического элементов массива меньшего заданного числа betta . В главной программе дано 3 одномерных массива a,b,c длиной 10 элементов каждый. Применить функцию для каждого из 3-х заданных массивов. (в функции не должно быть операторов ввода или вывода)

Задача 2.

1. Оформить функцию поиска суммы элементов, стоящих на нечетных местах (*использовать шаг цикла* $\neq 1$), В главной программе дано 3 одномерных массива a,b,c длиной 30 элементов каждый. Применить функцию для каждого из 3-х заданных массивов. Найти произведение найденных сумм элементов. (в функции не должно быть операторов ввода или вывода)

2. Оформить функцию поиска максимального элемента в одномерном массиве, В главной программе Дано 3 одномерных массива a,b,c длиной 20 элементов каждый. Применить функцию для каждого из 3-х заданных массивов. Найти сумму найденных максимальных элементов. (в функции не должно быть операторов ввода или вывода)

3. Оформить функцию поиска номера последнего нулевого элемента в массиве, В главной программе Дано 3 одномерных массива a,b,c длиной 20 элементов каждый. Применить функцию для каждого из 3-х заданных массивов. Найти сумму найденных номеров элементов. (в функции не должно быть операторов ввода или вывода)

4. Оформить функцию поиска минимального элемента среди положительных в одномерном массиве, В главной программе Дано 4 одномерных массива a,b,c,d

длиной 10 элементов каждый. применить функцию для каждого из 4-х заданных массивов. найти сумму найденных минимальных элементов. (в функции не должно быть операторов ввода или вывода)

5. Оформить функцию поиска номера последнего отрицательного элемента в массиве, В главной программе Дано 3 одномерных массива a,b,c длиной 20 элементов каждый. применить функцию для каждого из 3-х заданных массивов. найти разность найденных номеров элементов. (в функции не должно быть операторов ввода или вывода)

6. Оформить функцию поиска минимального элемента среди элементов, стоящих на четных местах ((*использовать шаг цикла* ≠1)), В главной программе Дано 4 одномерных массива a,b,c,d длиной 10 элементов каждый. применить функцию для каждого из 4-х заданных массивов. найти произведение найденных минимальных элементов. (в функции не должно быть операторов ввода или вывода)

7. Оформить функцию поиска номера последнего положительного элемента в массиве, В главной программе Дано 2 одномерных массива a,b,длиной 15 элементов каждый. применить функцию для каждого из 2-х заданных массивов. найти произведение найденных номеров элементов. (в функции не должно быть операторов ввода или вывода)

8. Оформить функцию поиска максимального элемента массива среди элементов, стоящих на нечетных местах ((*использовать шаг цикла* ≠1)),. В главной программе Дано 4 одномерных массива a,b,c,d длиной 10 элементов каждый. применить функцию для каждого из 4-х заданных массивов найти произведение найденных максимальных элементов. (в функции не должно быть операторов ввода или вывода)

9. Оформить функцию поиска номера максимального элемента массива среди элементов, стоящих на нечетных местах ((*использовать шаг цикла* ≠1)),. В главной программе Дано 3 одномерных массива a,b,c длиной 30 элементов каждый. применить функцию для каждого из 3-х заданных массивов найти произведение найденных номеров максимальных элементов. (в функции не должно быть операторов ввода или вывода)

10. Оформить функцию поиска произведения положительных элементов массива, В главной программе Дано 4 одномерных массива a,b,c,d длиной 10 элементов каждый. применить функцию для каждого из 4-х заданных массивов. найти сумму найденных произведений. (в функции не должно быть операторов ввода или вывода)

11. Оформить функцию поиска номера максимального элемента массива среди отрицательных элементов, В главной программе Дано 3 одномерных массива a,b,c длиной 30 элементов каждый. применить функцию для каждого из 3-х заданных массивов. найти произведение найденных номеров максимальных элементов. (в функции не должно быть операторов ввода или вывода)

12. Оформить функцию поиска номера минимального элемента среди положительных элементов массива, В главной программе Дано 2 одномерных массива a,b длиной 30 элементов каждый. применить функцию для каждого из 2-х заданных массивов. найти разность найденных номеров минимальных элементов. (в функции не должно быть операторов ввода или вывода)

13. Оформить функцию поиска суммы положительных элементов массива, В главной программе Дано 4 одномерных массива a,b,c,d длиной 10 элементов каждый, применить функцию для каждого из 4-х заданных массивов. найти произведение найденных сумм элементов. (в функции не должно быть операторов ввода или вывода)

14. Оформить функцию поиска суммы положительных элементов массива, стоящих на четных местах ((*использовать шаг цикла* ≠1)), В главной программе Дано 3

одномерных массива a,b,c длиной 30 элементов каждый , применить функцию для каждого из 3-х заданных массивов.. найти произведение найденных сумм элементов. (в функции не должно быть операторов ввода или вывода)

15. Оформить функцию поиска произведение положительных элементов массива, стоящих на четных местах ((*использовать шаг цикла* ≠1)), В главной программе Дано 3 одномерных массива a,b,c длиной 30 элементов каждый. применить функцию для каждого из 3-х заданных массивов. найти произведение найденных произведений элементов. (в функции не должно быть операторов ввода или вывода)

16. Оформить функцию поиска суммы квадратов элементов массива, В главной программе Дано 4 одномерных массива a,b,c,d длиной 10 элементов каждый, применить функцию для каждого из 4-х заданных массивов. найти произведение найденных сумм элементов. (в функции не должно быть операторов ввода или вывода)

Задача 3

1. Описать функцию, вычисляющую количество появлений заданного символа в заданной строке («заданные» – это входные параметры функции). В главной программе дано 2 строки символов S1 и S2. Выяснить, что больше количество символов '*' в строке S1 или количество символов '+' в строке S2, используя функцию.

2. Описать функцию, вычисляющую количество всех слов в заданной строке. В главной программе дано 3 строки символов S1 и S2 и S3. Найти количество всех слов в этих строках, используя функцию.

3. Описать функцию, вычисляющую количество слов «Иванушка» в тексте. В главной программе дано 3 текста S1 и S2 и S3. Выяснить, в каком тексте больше слов «Иванушка», используя функцию.

4. Описать функцию, вычисляющую количество цифр в тексте. В главной программе дано 2 строки символов S1 и S2. Выяснить, совпадает ли количество цифр в этих текстах, используя функцию.

5. Описать функцию, вычисляющую количество круглых скобок в тексте. В главной программе дано 2 текста S1 и S2 . Выяснить, в каком тексте больше скобок, используя функцию.

6. . Описать функцию, вычисляющую количество появлений заданного символа в заданной строке(«заданные» – это входные параметры функции).. В главной программе Дано 1 строка символов S1. Выяснить, совпадает ли количество круглых открывающихся скобок и круглых закрывающихся скобок в этом тексте, используя функцию.

7. Описать функцию, определяющую номер последней цифры в тексте. В главной программе дано 2 текста S1 и S2. Найти и вывести номера последних цифр в текстах.

8. Описать функцию, определяющую номер последнего символа равного заданному символу в заданном тексте («заданные» – это входные параметры функции).. В главной программе Дано 2 строки символов S1 и S2. Найти номер последнего символа «:» в S1 и номер последнего символа «;» в S2.

9. Описать функцию, вычисляющую количество латинских букв в тексте. В главной программе дано 2 текста S1 и S2 . Выяснить, в каком тексте больше латинских букв, используя функцию.

10. Описать функцию, вычисляющую количество появлений заданного символа в заданной строке(«заданные» – это входные параметры функции).. В главной

программе Дано 2 строки символов S1 и S2. Выяснить, что больше количество символов ’@’ в строке S1 или количество символов ‘$’ в строке S2, используя функцию.

11. Описать функцию, вычисляющую количество предложений в заданной строке. В главной программе дано 3 строки символов S1 и S2 и S3. Найти количество всех предложений в этих строках, используя функцию.

12. Описать функцию, вычисляющую количество квадратных скобок в тексте. В главной программе дано 2 текста S1 и S2 . Выяснить, в каком тексте больше скобок, используя функцию.

13. Описать функцию, определяющую номер последней закрывающейся скобки (круглой или квадратной) в тексте. В главной программе дано 2 текста S1 и S2. Найти и вывести номера последних скобок в текстах.

14. описать функцию, вычисляющую количество появлений заданного слова в заданной строке(«заданные» – это входные параметры функции).. В главной программе дано 2 строки символов s1 и s2. выяснить, что больше количество появлений слова ’Pascal’ в строке s1 или в строке s2, используя функцию.

15. Описать функцию, вычисляющую количество появлений заданного символа в заданной строке(«заданные» – это входные параметры функции).. В главной программе Дано 2 строки символов S1 и S2. Выяснить, что больше количество букв ’Z’ в строке S1 или в строке S2, используя функцию.

16. Описать функцию, вычисляющую количество появлений заданного слова в заданной строке(«заданные» – это входные параметры функции).. В главной программе Дано 1 строка символов S1. Выяснить, совпадает ли количество появлений слова ’Begin’ в строке S1 и количество появлений слова ’End’ , используя функцию.

17. Описать функцию, вычисляющую количество появлений заданного слова в заданной строке («заданные» – это входные параметры функции).. В главной программе Дано 2 строки символов S1 и S2. Выяснить, что больше количество появлений слова ’Иванов’ в строке S1 или количество появлений слова ’Петров’ в строке S2. ., используя функцию.

18. Описать функцию, вычисляющую количество появлений заданного символа в заданной строке («заданные» – это входные параметры функции).. В главной программе Дано 2 строки символов S1 и S2. Выяснить, что больше количество букв ’к’ в строке S1 или количество букв ‘н’ в строке S2, используя функцию.

19. Описать функцию, вычисляющую количество появлений заданного слова в заданной строке («заданные» – это входные параметры функции).. В главной программе Дано 2 строки символов S1 и S2. Выяснить, что больше количество появлений слова ’Информатика’ в строке S1 или количество появлений слова ’Технология’ в строке S2. ., используя функцию.

Задача 4

1. Создать комбинированный (структурный) тип для сведений о периодических изданиях (наименование издания, тираж, годовая стоимость). Описать функцию нахождения общей суммы стоимостей изданий в одном таком комбинированном массиве. Пользователь задает два комбинированных массива по N элементов в каждом.(для двух библиотек). Применить функцию два раза для заданных двух библиотек. (Так же будет уместно описать процедуру ввода комбинированного массива и процедуру вывода.)

2. Создать комбинированный (структурный) тип для меню детского кафе (наименование изделия, вес, стоимость). Описать функцию нахождения наименования самого дорого блюда дня в одном таком комбинированном массиве. Пользователь задает два комбинированных массива по N элементов в каждом. (два меню для разных дней). Применить функцию два раза для заданных двух меню. . (Так же будет уместно описать процедуру ввода комбинированного массива и процедуру вывода.)

3. Создать комбинированный (структурный) тип для меню ресторана "Дракон" (наименование изделия, вес, стоимость). Описать функцию нахождения общего веса изделий в одном таком комбинированном массиве. Пользователь задает два комбинированных массива по N элементов в каждом (для двух отделений ресторана). Применить функцию два раза для заданных двух отделений ресторана. . (Так же будет уместно описать процедуру ввода комбинированного массива и процедуру вывода.)

4. Создать комбинированный (структурный) тип для списка CD-дисков (название альбома, исполнитель, стиль, год выпуска, длительность, стоимость). Описать функцию нахождения общей длительности всех музыки на всех дисках в коллекции. Пользователь задает два комбинированных массива по N элементов в каждом.(для двух коллекций). Применить функцию два раза для заданных двух коллекций. . (Так же будет уместно описать процедуру ввода комбинированного массива и процедуру вывода.)

5. Создать комбинированный (структурный) тип для списка CD-дисков (название альбома, исполнитель, стиль, год выпуска, длительность, стоимость). Описать функцию нахождения количества дисков с указанным исполнителем в коллекции. Пользователь задает два комбинированных массива по N элементов в каждом.(для двух коллекций). Применить функцию два раза для заданных двух коллекций. . (Так же будет уместно описать процедуру ввода комбинированного массива и процедуру вывода.)

6. Создать комбинированный (структурный) тип для анкетные данные студентов (Ф. И. О., год рождения, адрес, сведения о родителях, средний балл). Описать функцию нахождения лучшего студента в группе (по среднему баллу). Пользователь задает два комбинированных массива по N элементов в каждом.(для двух групп). Применить функцию два раза для заданных двух групп. . (Так же будет уместно описать процедуру ввода комбинированного массива и процедуру вывода.)

7. Создать комбинированный (структурный) тип для расписание полетов самолетов (пункт посадки, время отправления, время прибытия, время полета, стоимость билета). Описать функцию нахождения самого короткого полета в одном таком комбинированном массиве. Пользователь задает два комбинированных массива по N элементов в каждом.(для двух аэропортов). Применить функцию два раза для заданных двух аэропортов. . (Так же будет уместно описать процедуру ввода комбинированного массива и процедуру вывода.)

8. Создать комбинированный (структурный) тип для списка CD-дисков (название альбома, исполнитель, стиль, год выпуска, длительность, стоимость). Описать функцию нахождения самого дорого диска в коллекции. Пользователь задает два комбинированных массива по N элементов в каждом.(для двух коллекций). Применить функцию два раза для заданных двух коллекций. . (Так же будет уместно описать процедуру ввода комбинированного массива и процедуру вывода.)

9. Создать комбинированный (структурный) тип для списка CD-дисков (название альбома, исполнитель, стиль, год выпуска, длительность, стоимость). Описать функцию нахождения количества дисков не старше указанного года в коллекции. Пользователь задает два комбинированных массива по N элементов в каждом.(для двух коллекций). Применить функцию два раза для заданных двух коллекций. . (Так же будет уместно описать процедуру ввода комбинированного массива и процедуру вывода.)

10. Создать комбинированный (структурный) тип для перечень товаров, имеющихся в продаже в магазине "Океан" (наименование, единица измерения, цена, количество). Описать функцию нахождения общей суммы стоимостей (цена*количество) в одном таком комбинированном массиве. Пользователь задает два комбинированных массива по N элементов в каждом.(для двух магазинов). Применить функцию два раза для заданных двух магазинов. . (Так же будет уместно описать процедуру ввода комбинированного массива и процедуру вывода.)

11. Создать комбинированный (структурный) тип для списка CD-дисков (название альбома, исполнитель, стиль, год выпуска, длительность, стоимость). Описать функцию нахождения количества дисков с указанным годом в коллекции. Пользователь задает два комбинированных массива по N элементов в каждом.(для двух коллекций). Применить функцию два раза для заданных двух коллекций. . (Так же будет уместно описать процедуру ввода комбинированного массива и процедуру вывода.)

12. Создать комбинированный (структурный) тип для график отпусков (Ф. И. О., дата начала отпуска, дата выхода на работу, количество дней). Описать функцию нахождения самого короткого отпуска в одном таком комбинированном массиве. Пользователь задает два комбинированных массива по N элементов в каждом.(для двух организаций). Применить функцию два раза для заданных двух организаций. . (Так же будет уместно описать процедуру ввода комбинированного массива и процедуру вывода.)

13. Создать комбинированный (структурный) тип для информацию о тестируемом (Ф.И.О., IQ, возраст). Описать функцию нахождения фамилии самого умного в группе. Пользователь задает два комбинированных массива по N элементов в каждом.(для двух групп). Применить функцию два раза для заданных двух групп. . (Так же будет уместно описать процедуру ввода комбинированного массива и процедуру вывода.)

14. Создать комбинированный (структурный) тип для списка CD-дисков (название альбома, исполнитель, стиль, год выпуска, длительность, стоимость). Описать функцию нахождения количества дисков с указанным стилем в коллекции. Пользователь задает два комбинированных массива по N элементов в каждом.(для двух коллекций). Применить функцию два раза для заданных двух коллекций. . (Так же будет уместно описать процедуру ввода комбинированного массива и процедуру вывода.)

15. Создать комбинированный (структурный) тип для списка CD-дисков (название альбома, исполнитель, стиль, год выпуска, длительность, стоимость). Описать функцию нахождения общей суммы стоимостей дисков в коллекции. Пользователь задает два комбинированных массива по N элементов в каждом.(для двух коллекций). Применить функцию два раза для заданных двух коллекций. . (Так же будет уместно описать процедуру ввода комбинированного массива и процедуру вывода.)

Задача 5

1. Описать функции нахождения min в одномерном массиве и функцию нахождения max в одномерном массиве. Дано 3 одномерных массива a,b,c длиной 20 элементов каждый. Использовать функции для облегчения решения задачи. Вычислить

$$t = \begin{cases} \min(b_i) * \max(a_i + c_i), & \text{если } \min(a_i) < \max(b_i) \\ \dfrac{\min(b_i + c_i)}{\min(a_i)} + \max(a_i), & \text{иначе} \end{cases}$$

где min(a_i) – означает наименьший элемент из массива a.
max(a_i) – означает наибольший элемент из массива a.

2. Описать функции нахождения min в одномерном массиве и функцию нахождения max в одномерном массиве. Дано 3 одномерных массива x,y,d длиной 40 элементов каждый. Использовать функции для облегчения решения задачи. Вычислить

$$t = \begin{cases} \min(d_i) + \max(x_i * y_i), & \text{если } \min(x_i) < \max(y_i) \\ \dfrac{\min(d_i + x_i)}{\min(d_i)} + \max(x_i), & \text{иначе} \end{cases}$$

где min(a_i) – означает наименьший элемент из массива a.
max(a_i) – означает наибольший элемент из массива a.

3. Описать подпрограмму-функцию для *вычисления суммы кубов элементов массива* – $\sum_{i=1}^{40} y_i^3$ для облегчения решения задачи: Дано 3 одномерных массива x,y,d длиной 40 элементов каждый. Вычислить

$$u = \begin{cases} \sum_{i=1}^{40} x_i^3 - \sum_{i=1}^{40} d_i^3, & \text{если } \sum_{i=1}^{40} y_i^3 > 0 \\ \sum_{i=1}^{40} y_i^3 / \sum d_i^3, & \text{иначе} \end{cases}$$

где $\sum_{i=1}^{40} y_i^3 = y_1^3 + y_2^3 + y_3^3 + ... + y_{40}^3$ - означает сумма кубов элементов массива *y*

4. Описать подходящую подпрограмму-функцию вычисления суммы элементов массива $\sum_{i=1}^{10} y_i$, для облегчения решения задачи: Дано 3 одномерных массива a,b,d длиной 10 элементов каждый. Вычислить

$$v = \begin{cases} \sum_{i=1}^{10} a_i - \sum_{i=1}^{10} d_i, & \text{если } \sum_{i=1}^{10} d_i > 0 \\ \sum_{i=1}^{10} d_i / \sum_{i=1}^{10} b_i, & \text{иначе} \end{cases}$$

где $\sum_{i=1}^{10} y_i = y_1 + y_2 + y_3 + ... + y_{10}$ - означает сумма элементов массива y

5. Описать подходящую подпрограмму-функцию (*вычисления суммы элементов массива*-– $\sum_{i=1}^{10} y_i$) для облегчения решения задачи: Дано 3 одномерных массива a,b,d длиной 10 элементов каждый. Вычислить

$$v = \begin{cases} (\sum_{i=1}^{10} a_i + 1) / \sum_{i=1}^{10} d_i, & \text{если } \sum_{i=1}^{10} d_i > 0 \\ (\sum_{i=1}^{10} a_i + \sum_{i=1}^{10} b_i) \sum_{i=1}^{10} b_i, & \text{иначе} \end{cases}$$

где $\sum_{i=1}^{10} y_i = y_1 + y_2 + y_3 + ... + y_{10}$ - означает сумма элементов массива y

6. Описать подпрограмму-функцию *вычисления произведения элементов массива*– $\prod_{i=1}^{20} a_i$ для облегчения решения задачи. Дано 3 одномерных массива a,b,d длиной 20 элементов каждый. Вычислить

$$v = \begin{cases} \prod_{i=1}^{20} a_i - \prod_{i=1}^{20} (b_i + d_i), & \text{если} \prod_{i=1}^{20} a_i > \prod_{i=1}^{20} d_i \\ \prod_{i==1}^{20} (b_i - a_i) + \prod_{i=1}^{20} d_i, & \text{Иначе} \end{cases}$$

где $\prod_{i=1}^{20} a_i = a_1 \cdot a_2 \cdot a_3 \cdot \ldots \cdot a_{20}$ - означает произведение элементов массива a

7. . Описать подходящую подпрограмму-функцию (*вычисления суммы произведений элемента массива на число в степени* – $a_1x^{10} + a_2x^9 + a_3x^8 + \ldots + a_9x + a_{10}$) . . Дано 2 одномерных массива a,b длиной 10 элементов каждый и 2 числа x,y. Вычислить выражение, вызвав 3 раза созданную функцию

$$\frac{(a_1x^{10} + a_2x^9 + a_3x^8 + \ldots + a_9x + a_{10})^2 - (b_1y^{10} + b_2y^9 + b_3y^8 + \ldots + b_9y + b_{10})}{b_1(x+y)^{10} + b_2(x+y)^9 + \ldots + b_{10}}$$

8. Описать подходящую подпрограмму-функцию (*вычисления суммы произведений элемента массива на число в степени* – $a_1x^{10} + a_2x^9 + a_3x^8 + \ldots + a_9x + a_{10}$) для облегчения решения задачи. Дано 3 одномерных массива a,b,c длиной 10 элементов каждый и 3 числа x,y,z.. Вычислить выражение, вызвав 3 раза созданную функцию

$$\frac{(a_1x^{10} + a_2x^9 + a_3x^8 + \ldots + a_9x + a_{10})^5 - (b_1y^{10} + b_2y^9 + b_3y^8 + \ldots + b_9y + b_{10})^3}{(b_1(x+y)^{10} + b_2(x+y)^9 + \ldots + b_{10})^2 + (a_1z^{10} + a_2z^9 + \ldots + a_{10})}$$

9. Дано 3 одномерных массива a,b,c длиной 10 элементов каждый и 3 числа x,y,z. Описать подходящую подпрограмму-функцию (*вычисления суммы произведений элемента массива на число в степени*) для облегчения решения задачи. . Вычислить выражение, вызвав 3 раза созданную функцию

$$\frac{(a_1x^1 + a_2x^2 + a_3x^3 + \ldots + a_9x^9 + a_{10}x^{10})^2 + (b_1y^1 + b_2y^2 + b_3y^3 + \ldots + b_9y^9 + b_{10}y^{10})^3}{(b_1(x+y)^1 + b_2(x+y)^2 + \ldots + b_{10}(x+y)^{10})^2 + (a_1z^1 + a_2z^2 + \ldots + a_2z^{10})}$$

10 Описать подпрограмму-функцию вычисления скалярного произведения векторов *(массивов)* $(x,y) = x_1y_1+x_2y_2+\ldots x_ny_n$ для облегчения решения задачи: Дано 3 одномерных массива (вектора) a,b,c длиной 10 элементов каждый и 3 числа x,y,z. Вычислить (a,b)*x+(b,c)*y-(a,c)*((a+c),b)*z. (Где (a,b) – обозначает скалярное произведение векторов)

11. Описать подпрограмму-функцию вычисления скалярного произведения векторов *(массивов)* $(x,y) = x_1y_1+x_2y_2+\ldots x_ny_n$ для облегчения решения задачи: Дано 3 одномерных массива (вектора) a,b,c длиной 20 элементов каждый и 3 числа x,y,z. Вычислить (a,c)*z+(a,b)*y-(a,(c+b))*((a-c),b)*z. (Где (a,b) – обозначает скалярное произведение векторов)

12. Описать подходящую подпрограмму-функцию (*вычисления суммы произведений элемента массива на число в степени*) для облегчения решения задачи:

Дано 3 одномерных массива a,b,c длиной 10 элементов каждый и 3 числа x,y,z. . Вычислить выражение, вызвав 3 раза созданную функцию $\frac{(a_1x^1+a_2x^2+a3x^3+...+a_9x^9+a_{10}x^{10})^2+(b_1y^1+b_2y^2+b_3y^3+...+b_9y^9+b_{10}y^{10})^3}{(b_1(x+y)^1+b_2(x+y)^2+...+b_{10}(x+y)^{10})^2}$

13. Описать подпрограмму-функцию *вычисления произведения элементов массива* – $\prod_{i=1}^{20} a_i$ для облегчения решения задачи: Дано 3 одномерных массива a,b,d длиной 20 элементов каждый. Вычислить

$$v=\begin{cases}\prod_{i=1}^{20}a_i-5\prod_{i=1}^{20}(b_i+d_i), & если \prod_{i=1}^{20}a_i>\prod_{i=1}^{20}d_i\\ 7\prod_{i==1}^{20}(b_i-a_i)/\prod_{i=1}^{20}d_i, & Иначе\end{cases}$$

где $\prod_{i=1}^{20} a_i = a_1 \cdot a_2 \cdot a_3 \cdot .... \cdot a_{20}$ - произведение элементов массива a.

14. Описать подпрограмму-функцию *вычисления суммы кубов элементов массива* – $\sum_{i=1}^{40} y_i^{\,3}$ для облегчения решения задачи. Дано 3 одномерных массива x,y,d длиной 10 элементов каждый. Вычислить

$$u=\begin{cases}\sum_{i=1}^{10}x_i^{\,3}+3\sum_{i=1}^{10}d_i^{\,3}, & если \sum_{i=1}^{10}yi^3>0\\ 8\sum_{i=1}^{10}yi^3*\sum_{i=1}^{10}di^3, & иначе\end{cases}$$

где $\sum_{i=1}^{40} y_i^{\,3} = y_1^3 + y_2^3 + y_3^3 + ... + y_{40}^3$ - сумма кубов элементов массива y.

15. Описать подходящую подпрограмму-функцию *вычисления суммы произведений элемента массива на число в степени* $-\ a_1x^{10}+a_2x^9+a_3x^8+...+a_9x+a_{10}$ для облегчения решения задачи: Дано 3 одномерных массива a,b,c длиной 10 элементов каждый и 3 числа x,y,z. . Вычислить выражение, вызвав 3 раза созданную функцию

$$\frac{(a_1x^{10}+a_2x^9+a_3x^8+...+a_9x+a_{10})^5-(b_1y^{10}+b_2y^9+b_3y^8+...+b_9y+b_{10})^3}{(b_1(x+y)^{10}+b_2(x+y)^9+...+b_{10})^3+(a_1z^{10}+a_2z^9+...+a_{10})}.$$

ЛАБОРАТОРНАЯ РАБОТА №9. ФУНКЦИИ ТИПА VOID

Цель лабораторной работы

Разработка программ языке С++ с использованием функций, не возвращающих значений.

Методические указания

Функция типа **void** выполняет определенные действия, но не возвращает никакого значения. Может иметь аргументы, хотя они могут также отсут-ствовать.

При ее описании не требуется оператор **return()**, но его можно исполь-зовать как способ прервать (завершить) работу функции. Вызов функции типа **void** происходит с помощью отдельного оператора, состоящего из имени функции и списка фактических параметров. В следующем примере приведена функция типа **void**, в которой значение массива изменяется внутри функции и возвращается в главную программу, так как массив передается в функцию в качестве указателя на него.

Пример использования функции типа **void** *с изменением массива:*

Описать функцию, которая получает в качестве параметров два исходных массива **x** и **y** и присваивает третьему массиву **z** поэлементную разность массивов **x** и **y.** Применить процедуру для двух разных пар массивов **a** и **b**; **c** и **d.**

```
#include<stdio.h>
 #include<conio.h>
 #include<math.h>
 #include<iostream.h>
 #include<time.h>
 #include<stdlib.h>
 void raznos t(int *x, int *y, int *z, int n) //описание функции
 {      int i;
        for ( i = 0 ; i < n ; i + +)
        { z[i] = x[i] – y[i]; }
}

 void main()
{
 int i, a[5], b[5], c[5], d[5], n, k[5], m[5];
 cout<<"введите n"<<endl;
 cin >> n;
 for (i = 0 ; i < n ; i ++)
 {
        a[i]=rand()%20+1;
        b[i]=rand()%5;
        c[i]=rand()%5+1;
        d[i]=rand()%15;
}
 raznost (a, b, k, n);//вызов функции, и вычисление разности в массив k
 raznost (c, d, m, n);//второй вызов функции для массивов c и d
 for ( i = 0 ; i < n ; i + +)
{       cout << " разность a и b [ " << I << " ] = " << k[i] << endl;
        cout << " разность c и d [ " << I << " ] = " << m[i] << endl;
```

```
}
getch();
}
```

ОБЛАСТЬ ВИДИМОСТИ ПЕРЕМЕННЫХ. ЛОКАЛЬНЫЕ И ГЛОБАЛЬНЫЕ ПЕРЕМЕННЫЕ

Все величины, описанные внутри функции, являются локальными, областью их действия является функция. При выходе из функции значения локальных переменных теряются. Глобальные переменные видны во всех функциях, где не описаны локальные переменные с теми же именами. Рекомендуется, как можно реже использовать глобальные переменные и стремиться к тому, чтобы функции были максимально независимы.

ПЕРЕДАЧА ПАРАМЕТРОВ ФУНКЦИИ

Передача параметров функции происходит одним из трех способов:

- по значению;
- по ссылке;
- по указателю.

1 ПЕРЕДАЧА ПАРАМЕТРОВ ПО ЗНАЧЕНИЮ

Рассмотрим первый случай *передачи параметров по значению* на следующем примере функции, обменивающей значения двух переменных между собой:

Пример:

```
void change (int x, int y)
{       int temp = x; x = y; y = temp;
        cout<< "Внутри функции x = " << x;
        cout<< "Внутри функции y= " <<y;
}

void main()
{int a = 3, b = 5;
cout << "до вызова функции a = " << a; // до вызова функции a = 3
cout<< "до вызова функции b = " << b ; //до вызова функции b = 5
change (&a, &b); /*внутри функции a = x = 5, b = y = 3, значения изменились*/
cout<< "после вызова функции a = " << a; /*после вызова функции a = 5, значение не изменилось*/
cout<< " после вызова функции b = " << b ; /* после вызова функции b=3 значение не изменилось*/
}
```

В этом примере при вызове функции **change** формальному параметру **x** присваивается значение фактического параметра **a**, равного **3**. Так как **x** является локальной переменной функции **change**, то ее значение доступно только внутри функции. Если значение **x** будет изменено в функции, то новое ее значение не будет доступно в главной программе, то есть не повлияет на значение **a**.

Если необходимо, что бы изменённое значение более двух переменных было доступно в главной программе невозможно использовать оператор **return**, так как оператор **return** не может возвращать более одного значения, то для организации

функции, возвращающей более одного изменённого значения, используются способы передачи аргументов *по ссылке или по указателю*. Для этого формальный параметр функции описывается *как указатель*, а при вызове функции в главной программе в качестве фактического параметра указывается адрес переменной, получаемый с помощью *операции получения адреса* с символом **&** «амперсанд» (см. гл. «4.1 Указатели, адреса и ссылки»**)**

2 ПЕРЕДАЧА ПАРАМЕТРОВ ПО УКАЗАТЕЛЮ

Рассмотрим второй случай *передачи параметров по указателю* на примере этой же функции обменивающей значения двух переменных, но будем уже использовать указатели на переменные.

Пример:

```
void change (int *x, int *y)
{       int temp = *x; *x = *y; *y = temp;
        cout<< "Внутри функции x = " << x*;
        cout<< "Внутри функции y= " <<y*;
}

void main()
{int a = 3, b = 5;
cout << "до вызова функции a = " << a; // до вызова функции a = 3
cout<< "до вызова функции b = " << b ; //до вызова функции b = 5
change (&a, &b); // внутри функции a = x = 5, b = y = 3
cout<< "после вызова функции a = " << a; //после вызова функции a=5
cout<<" после вызова функции b = " << b ;//после вызова функции b=3
}
```

При вызове функции указателю ***x** присваивается адрес переменной **a** (**x = &a**), а не значение переменной как в первом случае. Внутри функции происходит запись нового значения не в локальную переменную, а по указанному адресу внешней переменной **a**, что приводит к изменению значения переменной **a,** в главной функции.

3 ПЕРЕДАЧА ПАРАМЕТРОВ ПО ССЫЛКЕ

Рассмотрим третий случай *передачи параметров по ссылке* на примере этой же функции. Для передачи параметров по ссылке формальные параметры функции описываются как ссылки.

Пример:

```
void change ( int &x, int &y){int temp = x ; x = y ; y = temp;}
void main()
{
int a = 3, b = 5;
...change( a, b );...
}
```

При вызове функции формальному параметру-ссылке **x** присваивается значение переменной **a** (**x = a = 3**) и адрес переменной **a** (**&x = &a**) , что за собой повлечет изменение значения **a**, в главной функции.

Пример: Описать функцию, определяющую координаты центра тяжести (точку **O(x0,y0)**), системы материальных частиц с координатами $\mathbf{M_i(x_i,y_i)}$ и с массами $\mathbf{m_i}$, по следующим формулам:

$$x0 = \frac{\sum_{i=1}^{n} m_i x_i}{\sum_{i=1}^{n} m_i}; \qquad y0 = \frac{\sum_{i=1}^{n} m_i y_i}{\sum_{i=1}^{n} m_i};$$

В главной программе для двух заданных систем материальных частиц определить, совпадают ли их центры тяжести, если не то, вычислить расстояние между ними. Для каждой системы частиц заданы три массива x_i, y_i, m_i

```
#include<stdio.h>
 #include<conio.h>
 #include<math.h>
 #include<iostream.h>
 #include<time.h>
 #include<stdlib.h>
 void centr_k (int *x, int *y, int *m, int n, float &x0, float &y0)
 {
        float Smx = 0, Smy = 0, Sm = 0;
        for (int i=0;i<n;i++)
        {
              Smx + = m[i]*x[i]; Smy + = m[i]*y[i]; Sm + = m[i];
        }
 x0 = Smx / Sm;
 y0 = Smy / Sm;
}

 void main()
 {
 int m1[10], x1[10], y1[10];
 int m2[10], x2[10], y2[10];
 float x01, y01, x02, y02, O;
for (int i = 0; i< 10 ; i + +) {
        m1[i] = random(10);
        x1[i] = random(10)-5;
        y1[i] = random(15)-7;
        }
for (i = 0 ; i < 5 ; i + +) {
        m2[i] = random(10);
        x2[i] = random(10)-5;
        y2[i] = random(15)-7;
 }
 centr_k(x1, y1, m1 , 10 , x01 , y01);
 centr_k(x2 ,y2 ,m2 , 10 , x02 , y02);
 if ((x01 ! = x02) || (y01 ! = y02))
O = sqrt(pow(( x01 – x02 ), 2)+pow((y01 – y02),2));
 cout << "O=" << O;
}
```

Варианты заданий

Задача № 1

1. Создать функцию типа void с передачей параметров по ссылке или указателю, которая переводит время, заданное в минутах в секунды. . Для создаваемой функции: подобрать имя; выбрать имена и типы входных и выходных параметров; описать тело функции; в главной программе вызвать созданную подпрограмму более одного раза с различными входными данными. Вывести результаты в главной программе.
2. Создать функцию типа void с передачей параметров по ссылке или указателю, которая определяет периметр треугольника по трем его сторонам. Для создаваемой функции: подобрать имя; выбрать имена и типы входных и выходных параметров; описать тело функции; в главной программе вызвать созданную подпрограмму более одного раза с различными входными данными. Вывести результаты в главной программе.
3. Создать функцию типа void с передачей параметров по ссылке или указателю, которая возвращает номер квадранта, в котором находится точка. Для создаваемой функции: подобрать имя; выбрать имена и типы входных и выходных параметров; описать тело функции; в главной программе вызвать созданную подпрограмму более одного раза с различными входными данными. Вывести результаты в главной программе.
4. Создать функцию типа void с передачей параметров по ссылке или указателю, которая возвращает среднее арифметическое трех данных чисел. Для создаваемой функции: подобрать имя; выбрать имена и типы входных и выходных параметров; описать тело функции; в главной программе вызвать созданную подпрограмму более одного раза с различными входными данными. Вывести результаты в главной программе.
5. Создать функцию типа void с передачей параметров по ссылке или указателю, которая определяет площадь круга по его радиусу. Для создаваемой функции: подобрать имя; выбрать имена и типы входных и выходных параметров; описать тело функции; в главной программе вызвать созданную подпрограмму более одного раза с различными входными данными. Вывести результаты в главной программе.
6. Создать функцию типа void с передачей параметров по ссылке или указателю, которая возвращает остаток от деления двух натуральных чисел . Для создаваемой функции: подобрать имя; выбрать имена и типы входных и выходных параметров; описать тело функции; в главной программе вызвать созданную подпрограмму более одного раза с различными входными данными. Вывести результаты в главной программе.
7. Создать функцию типа void с передачей параметров по ссылке или указателю, которая переводит радианы в градусы. Для создаваемой функции: подобрать имя; выбрать имена и типы входных и выходных параметров; описать тело функции; в главной программе вызвать созданную подпрограмму более одного раза с различными входными данными. Вывести результаты в главной программе.
8. Создать функцию типа void с передачей параметров по ссылке или указателю, которая определяет длину отрезка по его координатам. Для создаваемой функции: подобрать имя; выбрать имена и типы входных и выходных параметров; описать тело функции; в главной программе вызвать созданную подпрограмму более

одного раза с различными входными данными. Вывести результаты в главной программе.

9. Создать функцию типа void с передачей параметров по ссылке или указателю, которая возвращает в долларах сумму, заданную в рублях. Для создаваемой функции: подобрать имя; выбрать имена и типы входных и выходных параметров; описать тело функции; в главной программе вызвать созданную подпрограмму более одного раза с различными входными данными. Вывести результаты в главной программе.
10. Создать функцию типа void с передачей параметров по ссылке или указателю, которая возвращает большее из двух данных чисел. Для создаваемой функции: подобрать имя; выбрать имена и типы входных и выходных параметров; описать тело функции; в главной программе вызвать созданную подпрограмму более одного раза с различными входными данными. Вывести результаты в главной программе.
11. Создать функцию типа void с передачей параметров по ссылке или указателю, которая определяет длину окружности по заданному радиусу. Для создаваемой функции: подобрать имя; выбрать имена и типы входных и выходных параметров; описать тело функции; в главной программе вызвать созданную подпрограмму более одного раза с различными входными данными. Вывести результаты в главной программе.
12. Создать функцию типа void с передачей параметров по ссылке или указателю, которая переводит скорость из км/час в м/сек. Для создаваемой функции: подобрать имя; выбрать имена и типы входных и выходных параметров; описать тело функции; в главной программе вызвать созданную подпрограмму более одного раза с различными входными данными. Вывести результаты в главной программе.
13. Создать функцию типа void с передачей параметров по ссылке или указателю, которая возвращает среднее геометрическое двух данных чисел. Для создаваемой функции: подобрать имя; выбрать имена и типы входных и выходных параметров; описать тело функции; в главной программе вызвать созданную подпрограмму более одного раза с различными входными данными. Вывести результаты в главной программе.
14. Создать функцию типа void с передачей параметров по ссылке или указателю, которая возвращает в рублях сумму, заданную в долларах. Для создаваемой функции: подобрать имя; выбрать имена и типы входных и выходных параметров; описать тело функции; в главной программе вызвать созданную подпрограмму более одного раза с различными входными данными. Вывести результаты в главной программе.
15. Создать функцию типа void с передачей параметров по ссылке или указателю, которая переводит градусы в радианы. Для создаваемой функции: подобрать имя; выбрать имена и типы входных и выходных параметров; описать тело функции; в главной программе вызвать созданную подпрограмму более одного раза с различными входными данными. Вывести результаты в главной программе.
16. Создать функцию типа void с передачей параметров по ссылке или указателю, которая возвращает меньшее из двух данных чисел. Для создаваемой функции: подобрать имя; выбрать имена и типы входных и выходных параметров; описать тело функции; в главной программе вызвать созданную подпрограмму более одного раза с различными входными данными. Вывести результаты в главной программе.

17. Создать функцию типа void с передачей параметров по ссылке или указателю, которая переводит время, заданное в секундах в минуты

Задача № 2

Вариант № 1 Реализовать функцию типа void с передачей параметров-результатов по ссылке. Функция вычисляет характеристики параллелограмма по заданным двум сторонам и углу между ними. Выходные результаты функции – характеристики параллелограмма – это периметр и площадь s= a b Cos α. В главной программе задано два параллелограмма. Найти их характеристики, вызвав функцию 2 раза.

Вариант №2 Реализовать функцию типа void с передачей параметров-результатов по ссылке. Функция вычисляет характеристики квадрата по заданной стороне. Выходные результаты функции – Характеристики квадрата – это периметр, диагональ и площадь. В главной программе задано два квадрата. Найти их характеристики, вызвав функцию 2 раза.

Вариант №3 Реализовать функцию типа void с передачей параметров-результатов по ссылке. Функция вычисляет характеристики прямоугольника по заданным двум сторонам . Выходные результаты функции – Характеристики прямоугольника – это периметр, диагональ и площадь . В главной программе задано два прямоугольника . Найти их характеристики, вызвав функцию 2 раза.

Вариант№ 4 Реализовать функцию типа void с передачей параметров-результатов по ссылке. Функция вычисляет характеристики равнобедренной трапеции по заданным двум основаниям a,b и высоте h. Выходные результаты функции Характеристики равнобедренной трапеции – это периметр, и площадь s= ½ (a +b) h..В главной программе задано две равнобедренные трапеции . Найти их характеристики, вызвав функцию 2 раза.

Вариант№ 5. Реализовать функцию типа void с передачей параметров-результатов по ссылке. Функция вычисляет характеристики окружности по заданному радиусу R. Выходные результаты функции Характеристики окружности – это длина окружности и площадь .В главной программе задано две окружности . Найти их характеристики, вызвав функцию 2 раза.

Вариант№ 6. Реализовать функцию типа void с передачей параметров-результатов по ссылке. Функция вычисляет характеристики сферы по заданному радиусу R. Выходные результаты функции – характеристики сферы – это площадь поверхности, $s=4 \pi R^2$ и объем сферы $v=4/3 \pi R^3$. .В главной программе задано две сферы. Найти их характеристики, вызвав функцию 2 раза.

Вариант№ 7. Реализовать функцию типа void с передачей параметров-результатов по ссылке. Функция вычисляет характеристики квадратной призмы по заданной высоте H и стороне основания a. Выходные результаты функции – характеристики квадратной призмы – это площадь основания и объем . В главной программе задано две квадратные призмы . Найти их характеристики, вызвав функцию 2 раза.

Вариант№ 8. Реализовать функцию типа void с передачей параметров-результатов по ссылке. Функция вычисляет характеристики правильной треугольной призмы по заданной высоте H и стороне основания a . Выходные результаты функции – характеристики правильной треугольной призмы – это площадь основания и объем . .В главной программе задано две треугольные призмы . Найти их характеристики, вызвав функцию 2 раза.

Вариант№9 Реализовать функцию типа void с передачей параметров-результатов по ссылке. Функция вычисляет характеристики цилиндра по заданной высоте H и радиусу основания R . Выходные результаты функции – характеристики цилиндра – это площадь основания и объем .В главной программе задано два цилиндра . Найти их характеристики, вызвав функцию 2 раза.

Вариант№10 Реализовать функцию типа void с передачей параметров-результатов по ссылке. Функция вычисляет характеристики цилиндра по заданной высоте H и радиусу основания R . Выходные результаты функции – характеристики цилиндра – это площадь боковой поверхности и объем . В главной программе задано два цилиндра . Найти их характеристики, вызвав функцию 2 раза.

Вариант№11 Реализовать функцию типа void с передачей параметров-результатов по ссылке. Функция вычисляет характеристики конуса по заданной высоте H и радиусу основания R . Выходные результаты функции – характеристики конуса – это площадь боковой поверхности S=π R $\sqrt{R^2+H^2}$ и объем v= 1/3 H π R^2, . .В главной программе задано два конуса . Найти их характеристики, вызвав функцию 2 раза.

Вариант№12 Реализовать функцию типа void с передачей параметров-результатов по ссылке. Функция вычисляет характеристики конуса по заданной высоте H и радиусу основания R . Выходные результаты функции – характеристики конуса – это площадь основания и объем v= 1/3 H π R^2. .В главной программе задано два конуса . Найти их характеристики, вызвав функцию 2 раза.

Вариант№13. Реализовать функцию типа void с передачей параметров-результатов по ссылке. Функция вычисляет характеристики Усеченного конуса по заданной высоте H и двум радиусам оснований R и r . Выходные результаты функции – характеристики Усеченного конуса – это площадь нижнего основания и объем v=1/3 H π (R^2 +r^2+R r), . .В главной программе задано два конуса . Найти их характеристики, вызвав функцию 2 раза.

Вариант№14 Реализовать функцию типа void с передачей параметров-результатов по ссылке. Функция вычисляет характеристики вектора на плоскости по заданным координатам x и y . Выходные результаты функции – характеристики вектора – это длина вектора и тангенс угла наклона (между осью OX и вектором) . .В главной программе задано два вектора . Найти их характеристики, вызвав функцию 2 раза.

Вариант№15. Реализовать функцию типа void с передачей параметров-результатов по ссылке. Функция вычисляет характеристики точки на плоскости по заданным координатам x и y . Выходные результаты функции – характеристики точки – это расстояние до начала координат и номер четверти, в которой находиться точка . .В главной программе задано две точки . Найти их характеристики, вызвав функцию 2 раза.

Задача № 3

1. Описать функцию типа void с передачей параметров по ссылке или указателю POWERA3(A,B), вычисляющую третью степень числа A и возвращающую ее в переменной B (A — входной, B — выходной параметр; оба параметра являются вещественными). С помощью этой функции найти третьи степени пяти данных чисел.

2. Описать функцию типа void с передачей параметров по ссылке или указателю, заменяющую в тексте слова «обязательно» на «возможно». В главной программе Дано 2 строки символов S1 и S2.. Применить функцию в главной программе для строк S1 и S2..

3. Описать функцию типа void с передачей параметров по ссылке или указателю ShiftLeft(A, B, C), выполняющую левый циклический .сдвиг: значение A переходит в C, значение C — в B, значение B — в A (A, B, C — вещественные параметры, являющиеся одновременно входными и выходными). С помощью этой функции в главной программе выполнить левый циклический сдвиг для двух данных наборов из трех чисел: (A1, B1, C1) и (A2, B2, C2),

4. Описать функцию типа void с передачей параметров по ссылке или указателю, заменяющую в тексте все малые буквы после точки и пробела на большие . В главной программе Дано 2 строки символов S1 и S2. Применить функцию для строк S1 и S2.

5. Описать функцию типа void с передачей параметров по ссылке или указателю POWERA234(A,B,C,D), вычисляющую вторую, третью и четвертую степень числа A и возвращающую эти степени соответственно в переменных B, C, D (A — входной, B, C, D — выходные параметры; все параметры являются вещественными). С помощью этой функции в главной программе найти вторую, третью и четвертую степень пяти данных чисел.

6. Описать функцию типа void с передачей параметров по ссылке или указателю**,** заменяющую в тексте слова «Pascal» на «C++». В главной программе Дано 3 строки символов S1 и S2 , S3. Применить функцию в главной программе для строк S1 и S2, S3.

7. Описать функцию типа void с передачей параметров по ссылке или указателю, заменяющую в тексте указанное слово на другое заданное слово . В главной программе Дано 2 строки символов S1 и S2. В тексте S1заменить слово «Пиноккио» на «Буратино», а в тексте S2 заменить слово «Мальвина» на «Марина», применяя функцию

8. Описать функцию типа void с передачей параметров по ссылке или указателю Меап(X, У, AMeап, GMeап), вычисляющую среднее арифметическое AMeап - (X+ У)/2 и среднее геометрическое GMeап = Sqrt(X*У) двух положительных чисел X и У (X и У — входные, AMeап и GMeап —- выходные параметры вещественного типа). С помощью этой функции в главной программе найти среднее арифметическое и среднее геометрическое для пар (A, B), (A, C), (A, D), если даны A, B, C, D.

9. Описать функцию типа void с передачей параметров по ссылке или указателю, заменяющую в тексте указанное слово на другое заданное слово . В главной программе Дано 2 строки символов S1 и S2. В тексте S1заменить слово «Грэтель» на «Василиса». В тексте S2 заменить слово «Синдерелла» на «Золушка», применяя функцию

10. Описать функцию типа void с передачей параметров по ссылке или указателю TrianglePS(a, P, S), вычисляющую по стороне a равностороннего треугольника его периметр P = 3*a и площадь $S = a^2*Sqrt(3)/4$ (a — входной, P и S— выходные параметры; все параметры являются вещественными). С помощью этой функции в главной программе найти периметры и площади трех равносторонних треугольников с данными сторонами.

11. Описать функцию типа void с передачей параметров по ссылке или указателю, заменяющую в тексте все малые буквы на большие . В главной программе Дано 2 строки символов S1 и S2. Применить функцию в главной программе для строк S1 и S2.

12. Описать функцию типа void с передачей параметров по ссылке или указателю RectPS(x1, y1,x2,y2, P, S), вычисляющую периметр P и площадь S прямоугольника со сторонами, параллельными осям координат, по координатам (x1, y1), (x2, y2) его противоположных вершин (x1, y1, X2, У2 — входные, P и S — выходные

параметры вещественного типа). С помощью этой функции в главной программе найти периметры и площади трех прямоугольников сданными противоположными вершинами,

13. Описать функцию типа void с передачей параметров по ссылке или указателю, заменяющую в тексте все большие буквы на малые . В главной программе Дано 3 строки символов S1 и S2 , S3.. Применить функцию в главной программе для строк S1 и S2, S3.

14. Описать функцию типа void с передачей параметров по ссылке или указателю Swap(X, У), меняющую содержимое переменных X и У (X и У — вещественные параметры, являющиеся одновременно входными и выходными). С ее помощью в главной программе для данных переменных A, B, C, D последовательно поменять содержимое следующих пар: A и B, C и D, B и C и вывести новые значения A, B, C, D.

15. Описать функцию типа void с передачей параметров по ссылке или указателю Mintax(X, У), записывающую в переменную X минимальное из значений X и У, а в переменную У — максимальное из этих значений (X и У — вещественные параметры, являющиеся одновременно входными и выходными). Используя четыре вызова этой функции в главной программе , найти минимальное и максимальное из данных чисел A, B, C, D.

16. Описать функцию типа void с передачей параметров по ссылке или указателю ShiftRight3(A,B, C), выполняющую правый циклический сдвиг: значение A переходит в B, значение B — в C, значение C — в A (A, B, C — вещественные параметры, являющиеся одновременно входными и выходными). С помощью этой функции в главной программе выполнить правый циклический сдвиг для двух данных наборов из трех чисел: (A1, B1, C1) и (A2,B2,C2).

Задача 4.

1. Описать подпрограмму-функцию типа void с передачей параметров по ссылке или указателю, переводящую декартовы координаты заданной точки (x,y) в полярные координаты по следующим формулам:  Входные параметры: (x,y), выходные параметры (по ссылке или указателю): ρ, φ.
В главной программе для двух заданных точек (x1,y1) и (x2,y2) найти их полярные координаты и найти сумму их углов $\varphi_1 + \varphi_2$ и разность радиус векторов $\rho 1 - \rho 2$.

2. Описать подпрограмму-функцию типа void с передачей параметров по ссылке или указателю, переводящую декартовы координаты заданной точки (x,y,z) в цилиндрические координаты по следующим формулам: $\rho = \sqrt{x^2 + y^2}$; $= arctg\left(\frac{y}{x}\right)$; $z_2 = z$ Входные параметры: (x,y,z), выходные параметры (по ссылке или указателю): ρ, φ, z2. В главной программе для двух заданных точек найти их цилиндрические координаты и найти сумму их углов $\varphi_1 + \varphi_2$ и разность радиус векторов $\rho 1 - \rho 2$.

3. Описать подпрограмму-функцию типа void с передачей параметров по ссылке или указателю, переводящую декартовы координаты заданной точки (x,y,z) в полярные координаты по следующим формулам:

$\rho=\sqrt{x^2+y^2+z^2};\ \varphi=arctg\left(\frac{y}{x}\right)$

$z=arctg\left(\frac{\sqrt{x^2+y^2}}{z}\right)$ Входные параметры: (x,y,z), выходные параметры (по ссылке или указателю): ρ, φ, z.
В главной программе для двух заданных точек найти их полярные координаты и найти сумму их углов $\varphi_1+\varphi_2$ и разность радиус векторов $\rho 1-\rho 2$.

4. Описать подпрограмму-функцию типа void с передачей параметров по ссылке или указателю, определяющую середину отрезка (точку O(x0,y0)), заданного координатами его концов (точки M1(x1,y1) M2(x2,y2)) следующим формулам: $x0=\frac{x1+x2}{2};\ y0=\frac{y1+y2}{2}$ Входные параметры: x1,y1,x2,y2 , выходные параметры (по ссылке или указателю): x0,y0.
В главной программе для двух заданных отрезков выяснить, не совпадают ли их середины.

5. Описать подпрограмму-функцию типа void с передачей параметров по ссылке или указателю, определяющую точку P(xp,yp), делящую заданный отрезок в отношении $\frac{1}{\lambda}$ (отрезок, задан координатами его концов (точки M1(x1,y1) M2(x2,y2)) по следующим формулам: $xp=\frac{x1+\lambda\cdot x2}{1+\lambda};\quad yp=\frac{y1+\lambda\cdot y2}{1+\lambda}$. Входные параметры: x1,y1,x2,y2,λ , выходные параметры (по ссылке или указателю): xp,yp . В главной программе для двух заданных отрезков выяснить не совпадают ли их вычисленные полюса.

6. Описать подпрограмму-функцию типа void с передачей параметров по ссылке или указателю, определяющую координаты центра тяжести (точку O(x0,y0)), системы материальных частиц с координатами $M_i(x_i,y_i)$ и с массами m_i . (должны быть заданы три массива x_i, y_i, m_i для задания одной системы частиц) по следующим формулам: $x0=\frac{\sum_{i=1}^{n}m_i x_i}{\sum_{i=1}^{n}m_i};\ \cdot\ y0=\frac{\sum_{i=1}^{n}m_i y_i}{\sum_{i=1}^{n}m_i};$ Входные параметры: три массива x_i, y_i, m_i, выходные параметры (по ссылке или указателю): x0,y0.
В главной программе для двух заданных систем материальных частиц выяснить не совпадают ли их центры тяжести.

7. Описать подпрограмму-функцию типа void с передачей параметров по ссылке или указателю, заполняющую одномерный массив x(n) элементами побочной диагонали матрицы A(n*n). Входные параметры: матрица A(n*n), выходные параметры (по ссылке или указателю): массив x(n) . В главной программе для двух заданных матриц C1(n*n) и C1(m*m) создать такие массивы и проверить равны ли их серединные элементы (т.е. элементы, расположенные в центре массива, если кол-во четное ,то ,на ваше усмотрение, левый или правый элемент от центра)

8. Описать подпрограмму-функцию типа void с передачей параметров по ссылке или указателю, переводящую декартовы координаты заданной точки (x,y) в полярные координаты по следующим формулам: $\rho=\sqrt{x^2+y^2};\ \varphi=arctg\left(\frac{y}{x}\right)$ Входные параметры: (x,y), выходные параметры (по ссылке или указателю): ρ, φ.

В главной программе для двух заданных точек найти их полярные координаты и найти сумму их углов $\varphi_1 + \varphi_2$ и разность радиус векторов $\rho 1 - \rho 2$.

9. Описать подпрограмму-функцию типа void с передачей параметров по ссылке или указателю, переводящую декартовы координаты заданной точки (x,y,z) в цилиндрические координаты по следующим формулам: $\rho = \sqrt{x^2 + y^2}$; $= arctg\left(\frac{y}{x}\right)$; $z_2 = z$. Входные параметры: (x,y,z), выходные параметры (по ссылке или указателю): ρ, φ, z2. В главной программе для двух заданных точек найти их цилиндрические координаты и найти сумму их углов $\varphi_1 + \varphi_2$ и разность радиус векторов $\rho 1 - \rho 2$.

10. Описать подпрограмму-функцию типа void с передачей параметров по ссылке или указателю, переводящую декартовы координаты заданной точки (x,y,z) в полярные координаты по следующим формулам: $\rho = \sqrt{x^2+y^2+z^2}$; $\varphi = arctg\left(\frac{y}{x}\right)$; $z = arctg\left(\frac{\sqrt{x^2+y^2}}{z}\right)$. Входные параметры: (x,y,z), выходные параметры (по ссылке или указателю): ρ, φ, z. В главной программе для двух заданных точек найти их полярные координаты и найти сумму их углов $\varphi_1 + \varphi_2$ и разность радиус векторов $\rho 1 - \rho 2$.

11. Описать подпрограмму-функцию типа void с передачей параметров по ссылке или указателю, определяющую середину отрезка (точку O(x0,y0)), заданного координатами его концов (точки M1(x1,y1) M2(x2,y2)) следующим формулам: $x0 = \frac{x1+x2}{2}$; $y0 = \frac{y1+y2}{2}$ Входные параметры: x1,y1,x2,y2 , выходные параметры (по ссылке или указателю): x0,y0. .
В главной программе для двух заданных отрезков выяснить, не совпадают ли их середины.

12. Описать подпрограмму-функцию типа void с передачей параметров по ссылке или указателю, определяющую точку P(xp,yp), делящую заданный отрезок в отношении $\frac{1}{\lambda}$ (отрезок, задан координатами его концов (точки M1(x1,y1) M2(x2,y2)) по следующим формулам: $xp = \frac{x1+x2\lambda}{1+\lambda}$; $yp = \frac{y1+y2\lambda}{1+\lambda}$. . Входные параметры: x1,y1,x2,y2,λ , выходные параметры (по ссылке или указателю): xp,yp .В главной программе для двух заданных отрезков выяснить не совпадают ли их вычисленные полюса.

13. Описать подпрограмму-функцию типа void с передачей параметров по ссылке или указателю, определяющую координаты центра тяжести (точку O(x0,y0)), системы материальных частиц с координатами $M_i(x_i,y_i)$ и с массами m_i . (должны быть заданы три массива x_i, y_i, m_i для задания одной системы частиц) по следующим формулам: $x0 = \frac{\sum_{i=1}^{n} m_i x_i}{\sum_{i=1}^{n} m_i}$; . $y0 = \frac{\sum_{i=1}^{n} m_i y_i}{\sum_{i=1}^{n} m_i}$; Входные параметры: три массива x_i, y_i, m_i, выходные параметры (по ссылке или указателю): x0,y0.
В главной программе для двух заданных систем материальных частиц выяснить не

совпадают ли их центры тяжести.

14. Описать подпрограмму-функцию типа void с передачей параметров по ссылке или указателю, заполняющую одномерный массив x(n) элементами побочной диагонали матрицы A(n*n). В главной программе для двух заданных матриц C1(n*n) и C1(m*m) создать такие массивы и проверить равны ли их серединные элементы (т.е. элементы, расположенные в центре массива, если кол-во четное ,то ,на ваше усмотрение, левый или правый элемент от центра)

Задача 5

1. Дано 3 квадратные матрицы A,B и C. Вычислить квадрат той матрицы (A*A), в которой след наименьший. Вычисление следа оформить в виде функции. Вычисление квадрата матрицы оформить в виде процедуры.(След матрицы – это сумма элементов главной диагонали)

2. Дано 3 квадратные матрицы A,B и C. Вычислить квадрат той матрицы (A*A), в которой сумма элементов побочной диагонали наибольшая. Вычисление суммы элементов оформить в виде функции. Вычисление квадрата матрицы оформить в виде процедуры

3. Дано 3 квадратные матрицы A,B и C. Вычислить квадрат той матрицы (A*A), в которой сумма элементов первого столбца наибольшая. Вычисление суммы элементов оформить в виде функции. Вычисление квадрата матрицы оформить в виде процедуры

4. Дано 3 квадратные матрицы A,B и C. Вычислить квадрат той матрицы (A*A), в которой сумма элементов последней строки наибольшая. Вычисление суммы элементов оформить в виде функции. Вычисление квадрата матрицы оформить в виде процедуры

5. Дано 3 квадратные матрицы A,B и C. Вычислить квадрат той матрицы (A*A), в которой сумма элементов третьей строки наименьшая. Вычисление суммы элементов оформить в виде функции. Вычисление квадрата матрицы оформить в виде процедуры

6. Дано 3 квадратные матрицы A,B и C. Вычислить квадрат той матрицы (A*A), в которой сумма элементов третьей строки наименьшая. Вычисление квадрата матрицы оформить в виде процедуры. Вычисление квадрата матрицы оформить в виде процедуры

7. Дано 3 квадратные матрицы A,B и C. Вычислить квадрат той матрицы (A*A), в которой количество нулевых элементов третьей строки наименьшее. Вычисление количества элементов оформить в виде функции. Вычисление квадрата матрицы оформить в виде процедуры

8. Дано 3 квадратные матрицы A,B и C. Вычислить квадрат той матрицы (A*A), в которой количество нулевых элементов побочной диагонали наибольшая. Вычисление количества элементов оформить в виде функции. Вычисление квадрата матрицы оформить в виде процедуры

9. Дано 3 квадратные матрицы A,B и C. Вычислить квадрат той матрицы (A*A), в которой количество нулевых элементов главной диагонали наибольшая. Вычисление количества элементов оформить в виде функции. Вычисление квадрата матрицы оформить в виде процедуры

10. Дано 3 квадратные матрицы A,B и C. Вычислить квадрат той матрицы (A*A), в которой произведение отрицательных элементов побочной диагонали наибольшая. Вычисление количества элементов оформить в виде функции. Вычисление квадрата матрицы оформить в виде процедуры

11. Дано 3 квадратные матрицы A,B и C. Вычислить квадрат той матрицы (A*A), в которой среднее арифметическое отрицательных элементов побочной диагонали наибольшая. Вычисление количества элементов оформить в виде функции. Вычисление квадрата матрицы оформить в виде процедуры

12. Дано 3 квадратные матрицы A,B и C. Вычислить квадрат той матрицы (A*A), в которой произведение положительных элементов первого столбца наибольшая. Вычисление суммы элементов оформить в виде функции. Вычисление квадрата матрицы оформить в виде процедуры

15. Дано 3 квадратные матрицы A,B и C. Вычислить квадрат той матрицы (A*A), в которой среднее арифметическое положительных элементов первого столбца наибольшая. Вычисление суммы элементов оформить в виде функции. Вычисление квадрата матрицы оформить в виде процедуры

13. Дано 3 квадратные матрицы A,B и C. Вычислить квадрат той матрицы (A*A), в которой произведение положительных элементов побочной диагонали наибольшая. Вычисление количества элементов оформить в виде функции. Вычисление квадрата матрицы оформить в виде процедуры

14. Дано 3 квадратные матрицы A,B и C. Вычислить квадрат той матрицы (A*A), в которой след наименьший. Вычисление следа оформить в виде функции. Вычисление квадрата матрицы оформить в виде процедуры.(След матрицы – это сумма элементов главной диагонали)

ЛАБОРАТОРНАЯ РАБОТА №10. ФУНКЦИИ, ВОЗВРАЩАЮЩИЕ МАССИВЫ

Цель лабораторной работы

Разработка программ языке С++ с использованием функций, возвращающие массивы данных

Методические указания

ФУНКЦИЯ, ВОЗВРАЩАЮЩАЯ МАССИВ

С помощью оператора **return** можно возвращать в главную программу не только значения простых переменных, но и указатели или массивы.

Пример:

Написать функцию, переставляющую исходный массив в обратном порядке и записывающую его в другой массив.

```
int * change( int *x, int n)
{int i;
        int *y = (int*) malloc (n*sizeof( int));
         for ( i = 0; i < n ; i + +)
                y [i] = x [n – i – 1];
return (y);
}

void main()
{ int *a, *b, *ac, *bc;
int na = 5, nb = 10;
a= (int*)malloc(na*sizeof(int));
b=(int*)malloc(nb*sizeof(int));
for(i = 0 ; i < na ; i + +)
        a[i] = rand()%10 - 5;
for(i = 0 ; i < nb ; i + +)
        b[i] = rand()%5 - 10;
ac = change (a, na);
for(i = 0 ; i < na ; i + +)
        cout << “a перестановка =” << ac[i];
bc = change (b, nb);
for(i = 0 ; i < nb ; i + + )
        cout << “b перестановка =” << bc[i];
}
```

Эту же программу можно оформить другим способом: так как массив является указателем и содержит адрес своего нулевого элемента, то, являясь параметром функции, массив будет изменяемым параметром. В следующем примере создадим функцию, которая переставляет массив в обратном порядке и записывает его в этот же массив.

```
#...
void change2 (int *x, int n)
{int i ;
for(i = 0 ; i < n / 2 ; i + +)
```

```
{       int temp = x[i];
        x[i] = x[n – i – 1];
        x[n – i – 1] = temp;}
}

void main()
{int *a, *ac;
int na = 5;
a = (int*)malloc(na*sizeof(int));
for(i = 0 ; i < n a ; i + + )
{       a[i] = random(10);
        cout << a[i];
}
change2 (a, na);
for(i = 0 ; i < na ; i + + )
        cout<< a[i];
}
```

ПАРАМЕТРЫ ФУНКЦИИ СО СПЕЦИФИКАТОРОМ CONST

В некоторых случаях требуется, чтобы параметр вернулся в главную программу из функции без изменений. Для того, чтобы защитить массив или любой другой параметр от нежелательных изменений, в теле функции нужно указывать спецификатор **const** при описании параметра.

Варианты заданий

Задача 1

1. Создать функцию типа void для поиска min и его номера в массиве. В главной программе Даны три вектора (массива) x(n),y(n),z(n). Вызвать 3 раза функцию типа void. Далее Если наименьший элемент вектора ***x*** положителен и находится в о 2-ой половине этого вектора и если в векторе **z** наименьший элемент равен 8, тогда все элементы вектора **y** , предыдущие перед его наименьшим элементом, заменить на 1.

2. Описать функцию типа void, которая присваивает параметру **x** квадрат меньшего числа из 2 –х данных вещественных чисел **x** и **y** , а параметру **y** квадрат большего из чисел **x** и **y.** Применить функцию типа void для трех разных пар чисел a и b; c и d ; p и q.

3. Создать функцию типа void для поиска max и его номера в массиве . В главной программе Даны три вектора(массива) x(n),y(n),z(n). Вызвать 3 раза функцию типа void. Далее Если наибольший элемент вектора **x** равен 7 и находится во 2-ой половине этого вектора и если в векторе **y** наибольший элемент равен 100, тогда все элементы вектора **z** , последующие за его наибольшим элементом, заменить на их модули.

4. Дано два одномерных массива **x** и **y** одинаковой длины **n**. Описать функцию типа void, которая присваивает третьему массиву **z** сумму векторов (массивов) **x** и **y.** Применить функцию типа void для двух разных пар массивов a и b; c и d.

5. Создать функцию типа void для поиска min и его номера в массиве . В главной программе Даны три вектора (массива) x(n),y(n),z(n). Вызвать 3 раза функцию типа void. Если наименьший элемент вектора **x** положителен и находится в о 2-ой половине этого вектора и если в векторе **y** наименьший элемент отрицателен, тогда все элементы вектора **z** , последующие за его наименьшим элементом, заменить число 100.

6. Дано два одномерных массива **x** и **y** одинаковой длины **n**. Описать функцию типа void, которая присваивает третьему массиву **z** сумму квадратов векторов (массивов) **x** и **y.** Применить функцию типа void для двух разных пар массивов a и b; c и d.

7. Создать функцию типа void для поиска max и его номера в массиве . В главной программе Даны три вектора (массива) x(n),y(n),z(n). Вызвать 3 раза функцию типа void. Если наибольший элемент вектора **x** отрицателен и находится в первой половине этого вектора и если в векторе **y** наибольший элемент равен 10, тогда все элементы вектора **z ,** предшествующие его наибольшему элементу, заменить на их квадраты,

8. .Дано два одномерных массива **x** и **y** одинаковой длины **n**. Описать функцию типа void, которая присваивает третьему массиву **z** разность векторов (массивов) **x** и **y.** Применить функцию типа void для двух разных пар массивов a и b; c и d.

9. . Дано два одномерных массива **x** и **y** одинаковой длины **n**. Описать функцию типа void, которая присваивает третьему массиву **z** по элементное произведение векторов (массивов) **x** и **y.** Применить функцию типа void для двух разных пар массивов. a и b; c и d

10. Переменной t присвоить значение true, если уравнения $kx^2+3.5x+a^2=0$ и $2x^2+7x+k+2=0$ имеют вещественные корни и при этом оба корня первого уравнения больше чем оба корнями второго. Переменной t присвоить значение false во всех остальных случаях. Описать функцию типа void для решения квадратного уравнения вида $ax^2+bx+c=0$.

11. Создать функцию типа void для поиска min и его номера в массиве . В главной программе Даны три вектора (массива) x(n),y(n),z(n). Вызвать 3 раза функцию типа void. Если наименьший элемент вектора **x** положителен и находится в первой половине этого вектора и если в векторе **y** наименьший элемент больше 10, тогда все элементы вектора **z** , последующие за его наименьшим элементом, заменить на их модули.

12. Дано два одномерных массива **x** и **y** одинаковой длины **n**. Описать функцию типа void, которая присваивает третьему массиву **z** сумму векторов (массивов) **x** и **y.** Использовать созданную функцию типа void для вычисления **d=a+b+c.**

13. Переменной t присвоить значение true, если уравнения $x^2+6.2x+a^2=0$ и $x^2+ax+b-1=0$ имеют вещественные корни и при этом оба корня первого уравнения лежат между корнями второго. Переменной t присвоить значение false во всех остальных случаях. Описать функцию типа void для решения квадратного уравнения вида $ax^2+bx+c=0$.

14. Создать функцию типа void для поиска max и его номера в массиве . В главной программе Даны три вектора x(n),y(n),z(n). Вызвать 3 раза функцию типа void. Если наибольший элемент вектора **x** равен 10 и находится в первой половине этого вектора и если в векторе **y** наибольший элемент отрицательный, тогда все элементы вектора **z** , предшествующие его наибольшему элементу, заменить на их кубы, .

15. . Создать функцию типа void для поиска min и его номера в массиве . В главной программе Даны три вектора (массива) x(n),y(n),z(n). Вызвать 3 раза функцию типа void. Если наименьший элемент вектора **x** равен 5 и находится в первой половине этого вектора и если в векторе **y** наименьший элемент больше 10, тогда все элементы вектора **z** , последующие за его наименьшим элементом, заменить на их квадраты. .

16. Описать функцию типа void, которая присваивает параметру **x** большее число из 2 –х данных вещественных чисел **x** и **y** , а параметру **y** меньшее из чисел **x** и **y.** Применить функцию типа void для трех разных пар чисел a и b; c и d ; p и q.

17. Описать функцию типа void, которая присваивает параметру **x** большее число из 3 –х данных вещественных чисел **x, y** и **z**, параметру **z** меньшее из чисел **x , y** и **z ,** а параметру **y** среднее из чисел **x , y** и **z.** Так что бы получилось **x > y > z.** Применить функцию типа void для двух разных троек чисел. a , b и c; d , p и q.

Задача 2

1. Описать две функции типа void сортировки одномерного массива длинной N по убыванию и по возрастанию Дано два массива a и в. Отсортировать ***a*** по возрастанию, ***в*** – по убыванию. Логической переменной t присвоить true, если наименьший элемент ***a*** больше наибольшего элемента ***в.***

2. Описать функцию типа void сортировки одномерного массива длинной N по убыванию методом «Выбора и перестановки .**max**».. В главной программе Дано два массива ***a*** и ***b***. Отсортировать ***a*** и, ***b*** – по убыванию. Поменять местами максимальный элемент массива ***a*** и второй наибольший (следующий за максимальным) в массиве ***b.*** Еще поменять местами второй наименьший массива **a** и наименьший массива ***b.***

3. Описать функцию типа void, сортировки одномерного массива длинной N по убыванию методом «Пузырька». В главной программе Дано два массива **a** и **b**. Отсортировать ***a*** и, ***b*** – по убыванию. Определить , что больше наименьший элемент ***a*** или наибольшей элемент ***b.***

4. Описать функцию типа void, сортировки одномерного массива длинной N по убыванию методом «Выбора и перестановки .**max**».. В главной программе Дано два массива ***a*** и ***b***. Отсортировать ***a*** и, ***b*** – по убыванию. Вывести квадрат наибольшего элемента массива ***a***, если второй наибольший (следующий за максимальным) в массиве ***b*** равен 20.

5. Описать функцию типа void, сортировки одномерного массива длинной N по возрастанию методом «Выбора и перестановки .**max**».. В главной программе Дано два массива ***a*** и ***b***. Отсортировать ***a*** и, ***b*** – по возрастанию. Поменять местами второй наименьший массива **a** и наибольший массива ***b.***

6. Описать функцию типа void, сортировки одномерного массива длинной N по убыванию методом «Пузырька». В главной программе Дано два массива ***a*** и ***b***. Отсортировать ***a*** и, ***b*** – по убыванию. Если максимальный элемент массива **b** отрицателен, то заменить все элементы массива ***a*** на их модули кроме первого(max) и последнего (min)

7. Описать две процедуры, сортировки одномерного массива длинной N по убыванию и по возрастанию Дано два массива x и y. Отсортировать x убыванию по, y– по возрастанию. Логической переменной t присвоить true, если наименьший элемент **y** больше наибольшего элемента ***x.***

8. Описать функцию типа void, сортировки одномерного массива длинной N по убыванию методом «Простой перестановки». В главной программе Дано два массива ***a*** и ***b***. Отсортировать ***a*** и, ***b*** – по убыванию. Если минимальный элемент массива **a** равен 1, то второй наибольший (следующий за максимальным) в массиве ***b*** заменить на 100.

9. Описать функцию типа void, сортировки одномерного массива длинной N по убыванию методом «Пузырька». В главной программе Дано два массива ***a*** и ***b***. Отсортировать ***a*** и, ***b*** – по убыванию. Если минимальный элемент массива **b** отрицателен, то заменить все элементы массива ***a*** на их модули.

10. Описать функцию типа void, сортировки одномерного массива длинной N по убыванию методом «Выбора и перестановки .**max**».. В главной программе Дано два массива ***a*** и ***b***. Отсортировать ***a*** и, ***b*** – по убыванию. Поменять местами минимальный элемент массива ***a*** и второй наибольший (следующий за максимальным) в массиве ***b***

11. . Описать функцию типа void, сортировки одномерного массива длинной N по возрастанию методом «Пузырька». В главной программе Дано два массива ***a*** и ***b***. Отсортировать ***a*** и, ***b*** – по возрастанию. Если максимальный элемент массива **a** <5, то заменить все элементы массива ***b*** на их квадраты кроме первого(min) и последнего (max).

12. Описать функцию типа void, сортировки одномерного массива длинной N по возрастанию методом «Выбора и перестановки .**max**».. В главной программе Дано два массива ***a*** и ***b***. Отсортировать ***a*** и, ***b*** – по возрастанию. Поменять местами второй наименьший массива **a** и наименьший массива ***b.***

13. Описать две процедуры, сортировки одномерного массива длинной N по убыванию и по возрастанию Дано два массива а и в. Отсортировать ***a*** по возрастанию, ***в*** – по убыванию. Логической переменной t присвоить true, если наименьший элемент ***a*** больше наибольшего элемента ***в.***

14. Описать функцию типа void, сортировки одномерного массива длинной N по возрастанию методом «Простой перестановки». В главной программе Дано два массива ***a*** и ***b***. Отсортировать ***a*** и, ***b*** – по возрастанию. Если минимальный элемент массива **a** <2, то заменить все элементы массива ***b*** на их квадраты.

15. Описать две процедуры, сортировки одномерного массива длинной N по убыванию и по возрастанию Дано два массива x и y. Отсортировать x убыванию по, y– по возрастанию. Логической переменной t присвоить true, если наименьший элемент **y** больше наибольшего элемента ***x.***

Задача № 3

1. Создать комбинированный (структурный) тип для расписания движения поездов (станция отправления, станция прибытия, время прибытия, время в пути). Описать процедуру сортировки комбинированного массива по названию станции отправления для одного массива. Пользователь задает два комбинированных массива по N элементов в каждом. (для электричек и поездов дальнего следования). Применить функцию два раза для заданных двух списков. . (Так же будет уместно описать процедуру ввода комбинированного массива и процедуру вывода.)
2. Создать комбинированный (структурный) тип для списка книг домашней библиотеки (автор, название книги, издательство, год издания, стоимость). Описать процедуру сортировки одного списка по году издания. Пользователь задает два комбинированных массива по N элементов в каждом (для двух списков книг). Применить функцию два раза для заданных двух списков книг. . (Так же будет уместно описать процедуру ввода комбинированного массива и процедуру вывода.)
3. Создать комбинированный (структурный) тип для расписания экзаменов и зачетов (предмет, вид отчетности, число, преподаватель). Описать процедуру сортировки по фамилии преподавателя в одном таком комбинированном массиве. Пользователь задает два комбинированных массива по N элементов в каждом.(для зимней и летней сессий). Применить функцию два раза для заданных двух сессий. . (Так же будет уместно описать процедуру ввода комбинированного массива и процедуру вывода.)
4. Создать комбинированный (структурный) тип для сведения о студентах (Ф. И. О., курс, группа, номер зачетки, средний балл). Описать процедуру сортировки по алфавиту по фамилии в одном таком комбинированном массиве. Пользователь задает два комбинированных массива по N элементов в каждом.(для двух университетов).

Применить функцию два раза для заданных двух университетов. . (Так же будет уместно описать процедуру ввода комбинированного массива и процедуру вывода.)

5. Создать комбинированный (структурный) тип для расписания учителя (номер урока, время начала урока, класс, предмет, номер кабинета). Описать процедуру сортировки по алфавиту по названию предмета в одном таком комбинированном массиве. Пользователь задает два комбинированных массива по N элементов в каждом.(для двух учителей). Применить функцию два раза для заданных двух учителей. . (Так же будет уместно описать процедуру ввода комбинированного массива и процедуру вывода.)

6. Создать комбинированный (структурный) тип для информации о наличии товаров на складе (наименование, артикул, дата получения, единица измерения, количество, цена). Описать процедуру сортировки по артикулу в одном комбинированном массиве. Пользователь задает два комбинированных массива по N элементов в каждом.(для двух складов). Применить функцию два раза для заданных двух складов. . (Так же будет уместно описать процедуру ввода комбинированного массива и процедуру вывода.)

7. Создать комбинированный (структурный) тип для "Телефонный справочник" (Ф. И. О., адрес, номер телефона). Описать процедуру сортировки по фамилии в одном комбинированном массиве. Пользователь задает два комбинированных массива по N элементов в каждом.(для двух пользователей). Применить функцию два раза для заданных двух пользователей. . (Так же будет уместно описать процедуру ввода комбинированного массива и процедуру вывода.)

8. Создать комбинированный (структурный) тип для списка CD-дисков (название альбома, исполнитель, стиль, год выпуска, длительность, стоимость). Описать процедуру сортировки CD-дисков по стоимости дисков в коллекции. Пользователь задает два комбинированных массива по N элементов в каждом.(для двух коллекций). Применить функцию два раза для заданных двух коллекций. . (Так же будет уместно описать процедуру ввода комбинированного массива и процедуру вывода.)

9. Создать комбинированный (структурный) тип для списка CD-дисков (название альбома, исполнитель, стиль, год выпуска, длительность, стоимость). Описать процедуру сортировки CD-дисков по имени исполнителя в коллекции. Пользователь задает два комбинированных массива по N элементов в каждом.(для двух коллекций). Применить функцию два раза для заданных двух коллекций. . (Так же будет уместно описать процедуру ввода комбинированного массива и процедуру вывода.)

10. Создать комбинированный (структурный) тип для списка CD-дисков (название альбома, исполнитель, стиль, год выпуска, длительность, стоимость). Описать процедуру сортировки CD-дисков по названию альбома в коллекции. Пользователь задает два комбинированных массива по N элементов в каждом.(для двух коллекций). Применить функцию два раза для заданных двух коллекций. . (Так же будет уместно описать процедуру ввода комбинированного массива и процедуру вывода.)

11. Создать комбинированный (структурный) тип для списка CD-дисков (название альбома, исполнитель, стиль, год выпуска, длительность, стоимость). Описать процедуру сортировки дисков по году выпуска в коллекции. Пользователь задает два комбинированных массива по N элементов в каждом. (для двух коллекций). Применить функцию два раза для заданных двух коллекций. . (Так же будет уместно описать процедуру ввода комбинированного массива и процедуру вывода.)

12. Создать комбинированный (структурный) тип для списка CD-дисков (название альбома, исполнитель, стиль, год выпуска, длительность, стоимость). Описать процедуру сортировки дисков по длительности в коллекции. Пользователь задает два

комбинированных массива по N элементов в каждом. (для двух коллекций). Применить функцию два раза для заданных двух коллекций. . (Так же будет уместно описать процедуру ввода комбинированного массива и процедуру вывода.)

13. Создать комбинированный (структурный) тип для списка книг домашней библиотеки (автор, название книги, издательство, год издания, стоимость). Описать процедуру сортировки одного списка по стоимости. Пользователь задает два комбинированных массива по N элементов в каждом (для двух списков книг). Применить функцию два раза для заданных двух списков книг. . (Так же будет уместно описать процедуру ввода комбинированного массива и процедуру вывода.)

14. Создать комбинированный (структурный) тип для списка книг домашней библиотеки (автор, название книги, издательство, год издания, стоимость). Описать процедуру сортировки одного списка по автору. Пользователь задает два комбинированных массива по N элементов в каждом (для двух списков книг). Применить функцию два раза для заданных двух списков книг. . (Так же будет уместно описать процедуру ввода комбинированного массива и процедуру вывода.)

15. Создать комбинированный (структурный) тип для списка книг домашней библиотеки (автор, название книги, издательство, год издания, стоимость). Описать процедуру сортировки одного списка по году издания. Пользователь задает два комбинированных массива по N элементов в каждом (для двух списков книг). Применить функцию два раза для заданных двух списков книг. . (Так же будет уместно описать процедуру ввода комбинированного массива и процедуру вывода.)

Задача №4

1. Описать функцию типа void поиска минимального элемента и его номера в заданной строке матрицы.(номер строки – входной параметр) В главной программе Даны две матрицы A(n□n), B(n□n). Поменять местами минимальные элементы в первых строках данных матриц.

2. Описать функцию типа void, меняющую местами q-ую и s-ую строки матрицы X(n*n)(параметры q и s входные параметры процедуры) Применить функцию типа void для обмена 2-ой и последней строки в матрице A и для обмена 3-ей и 5-ой строки в матрице B.

3. Описать функцию типа void поиска минимального элемента и его 2-ух позиций во всей матрице. Даны две матрицы A(n□n), B(n□n). Поменять местами минимальные элементы матриц A и B.

4. Даны три матрицы A(n*n), B(n*n), C(n*n) и три вектора x(n),y(n),z(n). Вычислить u=Ax+By–Cz+Bx. Использовать функцию типа void для вычисления произведения матрицы на вектор

5. Описать функцию типа void поиска максимального элемента и его номера в заданном столбце матрицы .(номер столбца – входной параметр). В главной программе Даны две матрицы A(n□n), B(n□n). Поменять местами максимальные элементы в первых столбцах данных матриц.

6. Описать функцию типа void отражения матрицы относительно главной диагонали. Даны две матрицы A(n□n), B(n□n)., Определить можно ли отражением относительно главной диагонали преобразовать одну в другую.

7. Описать функцию типа void поиска максимального элемента и его номера в заданной строке матрицы. .(номер строки – входной параметр) В главной программе Даны две матрицы A(n*m), B(n*m). Поменять местами максимальные элементы в 3-ой строке матрицы A и в 1-ей строке матрицы B.

8. Описать функцию типа void, меняющую местами первый и последний столбец матрицы X(n□n) Применить функцию типа void для двух разных матриц.

9. Заданы три матрицы **A** (n), **B**(n) и **C**(n). Выяснить и напечатать, сколько из них являются симметрическими. (Матрица называется симметрической, если транспонированная матрица равна исходной). Транспонирование матрицы оформить в виде процедуры.

10. Известно ,что матрицах может быть только один отрицательный элемент. Описать функцию типа void поиска отрицательного элемента и его местонахождения в матрице. В главной программе Даны две матрицы A(n*n), B(n*n). Поменять местами отрицательные элементы этих матриц

11. Описать функцию типа void поиска первого положительного элемента и его номера в заданном столбце матрицы. .(номер столбца – входной параметр) В главной программе Дана матрицы A(n*m). Поменять местами такие положительные элементы в первом и 3-ем столбцах матрицы.

12. Описать функцию типа void, меняющую местами к-ый и р-ый столбец матрицы X(n*n)(параметры k и p входные параметры) Применить функцию типа void для обмена 1-го и последнего столбца в матрице A и в матрице B.

13. Описать функцию типа void поиска последнего положительного элемента и его номера в заданной строке матрицы. .(номер строки – входной параметр). В главной программе Даны две матрицы A(n□n), B(n□n). Поменять местами такие положительные элементы в первой строке матрицы A и в 3-ей строке матрицы B.

14. Даны три матрицы A(n□n), B(n□n), C(n□n) и три вектора x(n),y(n),z(n). Вычислить u=Ax+Bz–Cy. Использовать функцию типа void для вычисления произведения матрицы на вектор.

15. Описать функцию типа void, меняющую местами p-ую и q-ую строки матрицы X(n*n)(параметры p и q входные параметры) Применить функцию типа void для обмена 1-ой и 3-ей строки в матрице A и для обмена 2-ой и последней строки в матрице B.

16. Описать функцию типа void поиска последнего отрицательного элемента и его номера в заданном столбце матрицы. .(номер столбца – входной параметр). В главной программе Даны две матрицы A(n*m), B(n*m). Поменять местами такие отрицательные элементы в 2-ом столбце матрицы A и в 4-ем столбце матрицы B.

17. Описать функцию типа void поиска максимального элемента и его 2-ух позиций во всей матрице. Даны две матрицы A(n□n), B(n□n). Поменять местами максимальные элементы матриц A и B.

18. Даны матрицы A(n□n), B(n□n) и два вектора x(n),y(n). Вычислить u=Ay+Bx . Использовать функцию типа void для вычисления произведения матрицы на вектор.

ЛАБОРАТОРНАЯ РАБОТА №11. РЕКУРСИЯ

Цель лабораторной работы

Разработка программ языке C++ с использованием рекурсивных функций.

Методические указания

Рекурсия достаточно распространённое явление, которое встречается не только в областях науки, но и в повседневной жизни. Например, эффект Дросте, треугольник Серпинского и т. д. Самый простой вариант увидеть рекурсию – это навести Web-камеру на экран монитора компьютера, естественно, предварительно её включив. Таким образом, камера будет записывать изображение экрана компьютера, и выводить его же на этот экран, получится что-то вроде замкнутого цикла. В итоге мы будем наблюдать нечто похожее на тоннель.

В программировании рекурсия тесно связана с функциями, точнее именно благодаря функциям в программировании существует такое понятие как рекурсия или рекурсивная функция. Простыми словами, рекурсия – определение части функции (метода) через саму себя, то есть это функция, которая вызывает саму себя, непосредственно (в своём теле) или косвенно (через другую функцию). Типичными рекурсивными задачами являются задачи: нахождения n!, числа Фибоначчи. Такие задачи мы уже решали, но с использованием циклов, то есть итеративно. Вообще говоря, всё то, что решается итеративно можно решить рекурсивно, то есть с использованием рекурсивной функции. Всё решение сводится к решению основного или, как ещё его называют, базового случая. Существует такое понятие как шаг рекурсии или рекурсивный вызов. В случае, когда рекурсивная функция вызывается для решения сложной задачи (не базового случая) выполняется некоторое количество рекурсивных вызовов или шагов, с целью сведения задачи к более простой. И так до тех пор пока не получим базовое решение. Разработаем программу, в которой объявлена рекурсивная функция, вычисляющая n!

Чтобы найти 5! нужно знать 4! и умножить его на 5; 4! = 4 * 3! и так далее. Согласно схеме, изображённой на рисунке 2, вычисление сведётся к нахождению частного случая, то есть 1!, после чего по очереди будут возвращаться значения каждому рекурсивному вызову. Последний рекурсивный вызов вернёт значение 5!.

Переделаем программу нахождения факториала так, чтобы получить таблицу факториалов. Для этого объявим цикл for, в котором будем вызывать рекурсивную функцию.

```
// factorial.cpp: определяет точку входа для консольного приложения.

#include "stdafx.h"
#include <iostream>
using namespace std;

unsigned long int factorial(unsigned long int);// прототип рекурсивной функции
int i = 1; // инициализация глобальной переменной для подсчёта кол-ва рекурсивных вызовов
unsigned long int result; // глобальная переменная для хранения возвращаемого результата рекурсивной функцией
```

```
int main(int argc, char* argv[])
{
    int n; // локальная переменная для передачи введенного числа с клавиатуры
    cout << "Enter n!: ";
    cin >> n;
    for (int k = 1; k <= n; k++ )
    {
     cout << k << "!" << "=" << factorial(k) << endl; // вызов рекурсивной функции
    }
    system("pause");
    return 0;
}

unsigned long int factorial(unsigned long int f) // рекурсивная функция для нахождения n!
{
    if (f == 1 || f == 0) // базовое или частное решение
        return 1; // все мы знаем, что 1!=1 и 0!=1
    //cout << "Step\t"<< i <<endl;
    i++;
    //cout <<"Result= "<< result << endl;
    result=f*factorial(f-1); // функция вызывает саму себя
    return result;
}
```

В строках 16 — 19 объявлен цикл, в котором вызывается рекурсивная функция. Всё ненужное в программе закомментированно. Запустив программу, нужно ввести значение, до которого необходимо вычислить факториалы. Результат работы программы показан на рисунке 3.

```
1!=1
2!=2
3!=6
4!=24
5!=120
6!=720
7!=5040
8!=40320
9!=362880
10!=3628800
11!=39916800
12!=479001600
13!=6227020800
14!=87178291200
```

Рисунок 3 — Рекурсия в C++

Теперь видно, насколько быстро возрастает факториал, кстати говоря, уже результат 14! не правильный, это и есть последствия нехватки размера типа данных. Правильное значение 14! = 87178291200.

Варианты заданий

Задача №1

1. Описать рекурсивную функцию для вычисления n –го члена ряда **3, 0.3, 0.03, 0.003, … 0.00003**

2. Описать рекурсивную функцию для вычисления n –го члена ряда **10, 5, 5/2, 5/4, 5/8, 5/16, …5/256**

3. Описать рекурсивную функцию для вычисления n –го члена ряда **6, 3, 3/2, ¾, 3/8, … 3/256**

4. Описать рекурсивную функцию для вычисления n –го члена ряда **1, 2, 4, 8, ….256**

5. Описать рекурсивную функцию для вычисления n –го члена ряда **2, 4, 6,….20**

6. Описать рекурсивную функцию для вычисления n –го члена ряда **1, 3, 5,….21**

7. Описать рекурсивную функцию для вычисления n –го члена ряда **1, 1.2, 1.4, 1.6, … 3.0**

8. Описать рекурсивную функцию для вычисления n –го члена ряда 0**.7 ; 0.07 ; 0.007 ; 0.0007; …0.00000000007**

9. Описать рекурсивную функцию для вычисления n –го члена ряда **1, 3, 9, 27, 81 ….729**

10. Описать рекурсивную функцию для вычисления n –го члена ряда **7, 17, 27, 37, …. 97**

11. Описать рекурсивную функцию для вычисления n –го члена ряда **3, 3.3, 3.6, 3.9, 4.2, ….6**

12. Описать рекурсивную функцию для вычисления n –го члена ряда **20, 10, 5, 5/2, 5/4, 5/8, 5/16, …5/128**

13. Описать рекурсивную функцию для вычисления n –го члена ряда **5,2; 5,4; 5,6; 5,8; 6; 6,2; … 10**

14. Описать рекурсивную функцию для вычисления n –го члена ряда **40, 20, 10, 5, 5/2, 5/4, … 5/128**

15. Описать рекурсивную функцию для вычисления n –го члена ряда **1, 4, 16, 64, 256, 1024, 4096**

Задача №2

1. .Описать рекурсивную функцию для вычисления произведения Вычислить :

$$y = \prod_{i=1}^{50} \lg(i + 0.5)$$

2. Описать рекурсивную функцию для вычисления произведения Вычислить :

$$y = \prod_{x=3}^{30} \sqrt{x+1}$$

3. Описать рекурсивную функцию для вычисления произведения Вычислить :

$$y = \prod_{x=2}^{15} (x^2 + \log_2 x)$$

4. Описать 2 рекурсивные функции для вычисления произведения и суммы . В главной программе вызвать эти функции для вычисления произведения :

$$y = m \prod_{k=3}^{8} \frac{1}{k} + \frac{1}{m} * \sum_{p=1}^{5} \frac{a^p}{\sin p}$$

5. Описать 2 рекурсивные функции для вычисления произведения и суммы . В главной программе вызвать эти функции для вычисления выражения :

$$y = \sqrt[3]{\prod_{x=3}^{9} \frac{2 \lg x}{x}} * \frac{35{,}29}{\sum_{j=1}^{7} j}$$

6. Описать 2 рекурсивные функции для вычисления произведения и суммы . В главной программе вызвать эти функции для вычисления выражения :

$$y = \sqrt[3]{\prod_{k=1}^{4} a^k * \sum_{j=6}^{9} a^{-j}}$$

7. Описать 2 рекурсивные функции для вычисления произведения и суммы . В главной программе вызвать эти функции для вычисления выражения :

$$y = \prod_{k=1}^{8} (k + 2^k) + \sum_{j=3}^{9} \sin 2j$$

8. Описать рекурсивную функцию для вычисления суммы Вычислить :

$$y = \sum_{x=2}^{10} e^{x + \log_2 x}$$

9. Описать рекурсивную функцию для вычисления произведения Вычислить :

$$y = \prod_{x=2}^{15} \frac{\ln x}{5}$$

10. Описать рекурсивную функцию для вычисления произведения Вычислить :

$$y = \prod_{x=3}^{30} 2\sqrt{x}$$

11. Описать рекурсивную функцию для вычисления суммы Вычислить :

$$y = \sum_{x=2}^{12} \log_5 x$$

12. Описать рекурсивную функцию для вычисления произведения Вычислить :

$$y = \prod_{x=3}^{25} 5 \sin 2x$$

13. Описать рекурсивную функцию для вычисления суммы Вычислить :

$$y = \sum_{x=1}^{15} \cos 5x * \sin 2x$$

14. Описать рекурсивную функцию для вычисления произведения Вычислить :

$$y = \prod_{n=1}^{12} n^x$$

Задача №3

1. Описать рекурсивную функцию для вычисления суммы первых k слагаемых. Входные параметры функции : x – вещественное и k – целое

$$S = x + \frac{x}{4} + \frac{x}{9} + \frac{x}{16} + \frac{x}{25} \ldots + \frac{x}{100}$$

2. Описать рекурсивную функцию для вычисления суммы первых k слагаемых. Входные параметры функции: x – вещественное и k – целое

$$S = \sin x + \sin 2x + \sin 3x + \ldots + \sin 7x$$

3. Описать рекурсивную функцию для вычисления суммы первых k слагаемых. Входные параметры функции: x – вещественное и k – целое

$$S = \cos x + 2\cos 2x + 3\cos 3x + \ldots + 6\cos 6x$$

4. Описать рекурсивную функцию для вычисления суммы первых k слагаемых. Входные параметры функции: x – вещественное и k – целое

$$S = \frac{e^{-x}}{2} + \frac{e^{-2x}}{3} + \frac{e^{-3x}}{4} + \ldots + \frac{e^{-8x}}{9}$$

5. Описать рекурсивную функцию для вычисления суммы первых k слагаемых. Входные параметры : функции x – вещественное и k – целое

$$S = \sin 3x + \sin 4x + \sin 5x + \ldots + \sin 11x$$

6. Описать рекурсивную функцию для вычисления суммы первых k слагаемых. Входные параметры функции: x – вещественное и k – целое

$$S = \cos 2x + \cos 3x + \cos 4x + \ldots + \cos 10x$$

7. Описать рекурсивную функцию для вычисления суммы первых k слагаемых. Входные параметры функции: x – вещественное и k – целое

$$S = 2x + \frac{3x}{2} + \frac{4x}{3} + \frac{5x}{4} + \ldots + \frac{11x}{10}$$

8. Описать рекурсивную функцию для вычисления суммы первых k слагаемых. Входные параметры функции: x – вещественное и k – целое

$$S = \frac{\sin x}{2} + \frac{\sin 2x}{3} + \frac{\sin 3x}{4} + \ldots + \frac{\sin 9x}{10}$$

9. Описать рекурсивную функцию для вычисления суммы первых k слагаемых. Входные параметры функции: x – вещественное и k – целое

$$S = \cos x + \cos 2x + \cos 3x + \ldots + \cos 13x$$

10. Описать рекурсивную функцию для вычисления суммы первых k слагаемых. Входные параметры функции: x – вещественное и k – целое

$$S = \sin x + \frac{\sin 2x}{4} + \frac{\sin 3x}{9} + \frac{\sin 4x}{16} \ldots + \frac{\sin 9x}{81}$$

11. Описать рекурсивную функцию для вычисления суммы первых k слагаемых. Входные параметры функции: x – вещественное и k – целое

$$S = \frac{\cos x}{2} + \frac{\cos 2x}{4} + \frac{\cos 3x}{6} + \ldots + \frac{\cos 8x}{16}$$

12. Описать рекурсивную функцию для вычисления суммы первых k слагаемых. Входные параметры функции: x – вещественное и k – целое

$$S = 2\cos x + 3\cos 2x + 4\cos 3x + \ldots + 8\cos 7x$$

13. Описать рекурсивную функцию для вычисления произведения первых n сомножителей.

p = (1 +2/2^3)(1 + 2 /3^3)(1 + 2/4^3)…(1 + 2/n^3)

14. Описать рекурсивную функцию для вычисления произведения n нечетных натуральных чисел

p = (2n – 1)!!=1*3*5*7*…*(2n-1)

15. Описать рекурсивную функцию для вычисления произведения n четных натуральных чисел

p = (2n)!!=2*4*6*8*…*(2n)

Задача №4

1. Описать рекурсивную процедуру для вычисления i –го члена ряда по формуле:

$x_i = \frac{x_{i-1}}{2} + \frac{a}{2 \cdot x_{i-1}}$, $x_1 = 1$, $a>0$

2. Описать рекурсивную функцию для вычисления n –го члена ряда по формуле:

$T_n=3\cdot T_{n-1}+2\cdot T_{n-2}$; $T_0=1$; $T_1=2$;

3. Описать рекурсивную функцию для вычисления n –го члена ряда по формуле:

$T_n=3\cdot T_{n-1}-\cdot T^5_{n-2}$; $T_0=2$; $T_1=2$;

4. Описать рекурсивную функцию для вычисления n –го члена ряда по формуле:

$R_n=2\cdot R_{n-1}-5\cdot R_{n-2}+R^2_{n-1}$; $R_0=1$; $R_1=1$;

5. Описать рекурсивную функцию для вычисления n –го члена ряда по формуле:

$R_n=(2\cdot R_{n-1}+R_{n-2})^3$; $R_0=2$; $R_1=1$;

6. Описать рекурсивную функцию для вычисления n –го члена ряда по формуле:

$R_n=(2\cdot R_{n-1}+4\cdot R_{n-2})\cdot 10+ R_{n-1}$; $R_0=5$; $R_1=3$;

7. Описать рекурсивную функцию для вычисления n –го члена ряда по формуле:

$k_n=12\cdot k_{n-1}+6\cdot k_{n-2}$; $k_0=1$; $k_1=5$;

8. Описать рекурсивную функцию для вычисления n –го члена ряда по формуле:

$z_n=2\cdot z_{n-1}+6\cdot z^2_{n-2}$; $z_0=5$; $z_1=1$

9. Описать рекурсивную функцию для вычисления n –го члена ряда по формуле:

$y_n=7\cdot y_{n-1}+3/y_{n-2}+15$; $y_0=1$; $y_1=2$;

10. Описать рекурсивную функцию для вычисления n –го члена ряда по формуле:

$y_n=3\cdot(y_{n-2}+3\cdot y^3_{n-1})+6$; $y_0=0$; $y_1=1$;

11. писать рекурсивную функцию для вычисления n –го члена ряда по формуле:

$z_n=a\cdot z_{n-2}+b\cdot z^5_{n-1}$; $z_0=5$; $z_1=1$

12. Описать рекурсивную функцию для вычисления n –го члена ряда по формуле:

$z_n=a\cdot z^2_{n-1}+b\cdot z_{n-2}$; $z_0=4$; $z_1=0$

13. Описать рекурсивную функцию для вычисления n –го члена ряда по формуле:

$y_n=7\cdot\sqrt{y_{n-1}}+3/y_{n-2}+15$; $y_0=1$; $y_1=2$;

14. Описать рекурсивную функцию для вычисления n –го члена ряда по формуле:

$y_n=3\cdot(\sqrt{y_{n-2}}+3\cdot y^3_{n-1})+6$; $y_0=0$; $y_1=1$;

Задача №5

1 Описать рекурсивную функцию для вычисления $y=\cos(15+\cos(14+\cos(...+\cos(2+\cos(1)))...))$

2 Описать рекурсивную функцию для вычисления $y=\sqrt{1+\sqrt{2+...+\sqrt{14+\sqrt{15}}}}$

3 Описать рекурсивную функцию для вычисления $y=\sqrt{x^5+\sqrt{x^4+...+\sqrt{x+1}}}$ (не использовать: $x^n=pow(x,n)$)

4 Описать рекурсивную функцию для вычисления $y=((...((\sin x^{25}+x)\sin x^{24}+x)\sin x^{23}+...+x)\sin^2+x)\sin x+x)$ X - произвольное число.

5 Описать рекурсивную функцию для вычисления $Y=\sqrt{3+\sqrt{6+\sqrt{9+\cdots+\sqrt{96+\sqrt{99}}}}}$

6 Описать рекурсивную функцию для вычисления $Y=\sqrt{99+\sqrt{96+\cdots+\sqrt{6+\sqrt{3}}}}$

7 Описать рекурсивную функцию для вычисления $Y=\sqrt{2+\sqrt{4+\sqrt{8+\cdots+\sqrt{512+\sqrt{1024}}}}}$

8 Описать рекурсивную функцию для вычисления $Y=\sqrt{1024+\sqrt{512+\cdots+\sqrt{4+\sqrt{2}}}}$

9 Описать рекурсивную функцию для вычисления $Y=\sqrt{SinX+\sqrt{Sin2X+\cdots+\sqrt{Sin98X+\sqrt{Sin99x}}}}$

10 Описать рекурсивную функцию для вычисления

$$Y = \sqrt{Sin99x + \sqrt{Sin98X + \cdots + \sqrt{Sin2X + \sqrt{SinX}}}}$$

11 Описать рекурсивную функцию для вычисления

$$Y = \sqrt{Sin(X + \sqrt{Sin(2X + \cdots + \sqrt{Sin(98X + \sqrt{Sin99X})\cdots))}}}$$

12 Описать рекурсивную функцию для вычисления

$$Y = \sqrt{Sin(X + \sqrt{Sin2(X + \cdots + \sqrt{Sin98(X + \sqrt{Sin99X})\cdots))}}}$$

13 Описать рекурсивную функцию для вычисления

$$Y = Sin\sqrt{X + Sin\sqrt{2X + \cdots + Sin\sqrt{98X + Sin\sqrt{99X}}}}$$

14 Описать рекурсивную функцию для вычисления

$$Y = Sin\sqrt{X + Sin2\sqrt{X + \cdots + Sin98\sqrt{X + Sin99\sqrt{X}}}}$$

ЛАБОРАТОРНАЯ РАБОТА №12. ФАЙЛОВЫЕ ПОТОКИ ВВОДА-ВЫВОДА

Цель лабораторной работы

Разработка программ языке С++ с использованием файловых потоков ввода-вывода.

Методические указания

РЕАЛИЗАЦИЯ РАБОТЫ С ФАЙЛАМИ

Под файлом обычно подразумеваются информация на внешнем носителе, например на жестком или гибком магнитном диске. Файл можно представить как конечное количество последовательных байтов. По способу доступа файлы можно разделить на *последовательные* и файлы с *произвольным* доступом. Различают также форматный и бесформатный файловый ввод/вывод.

Для реализации работы с внешними файлами необходимо подключить к программе заголовочный файл <**stdio.h**> или **<cstdio>.**

Перед открытием файла сначало необходимо описать указатель на предопределенную структуру типа **FILE** следующим образом:

FILE * имя_указателя_на_ файл;

Пример:

FILE *f1, *infile, *outfile;

После этого открыть файл либо для чтения либо для записи следующим образом:

указатель_файла = fopen(адрес_файла, режим_открытия);

Первый параметр функции **fopen** «адрес_файла» указывает текстовую строку, содержащую имя файла (маршрут доступа к файлу) , второй параметр определяет режим открытия.

При успешном открытии потока функция **fopen** возвращает указатель на файл, в противном случае возвращается значение **null**.

Режимы открытия могут быть следующими — :

- "r" — файл открывается только для чтения;
- "w" — файл открывается только для записи (если он уже существует, то стирается);
- "a" — файл открывается для добавления информации в его конец ;
- "r+" — файл открывается для чтения и записи (файл должен существовать);
- "w+" — открывается пустой файл для чтения и записи (если он уже существует, то стирается);
- "a+" — файл открывается для чтения и добавления информации в его конец ;
- Режим открытия может содержать символы **t** или **b, t** означает, что режим текстовый, **b** означает, что режим двоичный;

Пример:

f1 = fopen(“c:\cpp\data.txt”, ”rt”);

infile = fopen(“input.txt”, ”r+t”);

outfile = fopen(“output.txt”, ”a+t”);

Форматный ввод из потока осуществляется с помощью функции **fscanf(),** с дополнительным первым параметром указывающим имя файлового указателя.

Пример:

fscanf(infile, “%d”, &dat);

fscanf(f1, “%c”, &x);

Форматный вывод из потока осуществляется с помощью функции **fprintf();**

Пример:

fprintf (outfile, “значение=%d”, x);

fprintf (outfile , “sum=%f”, sum);

Бесформатный ввод/вывод строки из потока осуществляется с помощью функций **fgets();** и **fputs();**

Пример:

fgets (str, 20, intfile);// считывает из **intfile** 20 символов в строку **str**

fputs(str, outfile);//записывает в **outfile** строку **str**

По окончании работы с файлом его нужно закрыть функцией **fclose(указатель_файла); ,** например: **fclose(outfile);**

ФАЙЛОВЫЕ ПОТОКИ ВВОДА-ВЫВОДА

В языке С++ описаны стандартные библиотеки классов для работы с файлами. Стандартная библиотека содержит три класса для работы с файлами:

ifstream – класс входных файловых потоков;

ofstream – класс выходных файловых потоков;

fstream – класс двунаправленных файловых потоков;

Чтобы их использовать необходимо подключить к программе заголовочный файлы **<fstream>, <iostream>** и **<cstdlib>, а также**

Использование файлов в программе предполагает следующие операции:

1 создание потока;

2 открытие потока и связывание его с файлом;

3 обмен (ввод/вывод);

4 закрытие файла;

Опишем каждый из них:

1Создание потока происходи с помощью операторов **ifstream <имя_входного_потока>** или **ofstream<имя_выходного_потока>.**

Пример:

ifstream f1;

ifstream infile ;

ofstream outfile ;

2 Открытие потока и связывание его с файлом с помощью метода **open** классов : **<имя_потока>.open(“адрес_файла”);**

Пример:

f1.open (“c:\cpp\data.txt”);

infile.open (“input.txt”); ;

outfile .open (“output.txt”);

3 Обмен (ввод/вывод) происходит таким же образом как и ввод/вывод из стандартных потоков **cin** и **cout,** то есть с помощью перегруженных операций >> и <<.

Пример:

f1 >> x; или **f1.getline(str, 11)**

infile >> dat;

outfile << ”sum=”, sum; .

4 Закрытие файла с помощью метода **close:**

Пример:

```
f1 . close();
infile.close();
outfile.close();
```

Пример задачи, считывающие из файла исходные данные.

Определить комбинированный тип для представления анкеты ребенка, состоящей из его имени, пола и роста. Считать из заданного файла **kids.dat** информацию о детях. Вывести средний рост мальчиков.

```
#include <fstream.h>
#include <iostream.h>
#include <stdio.h>
#include <conio.h>
#include <string.h>
#include <stdlib.h>
#include <malloc.h>

typedef struct deti {
        char name[10];
        char pol[1];
        int rost;
} deti;

void main()
{

        float avg_rost;
        deti * children;//динамичесий массив
        int N, i;
        cout<<"Vvedite kolichestvo detei"<<endl;
        cin>>N;
        children = (deti*)malloc (N*sizeof(deti));
        ifstream f_children("C:/kids.dat");// открытие входного потока
        for (i=0; i<N; i++)
         {
                f_children>>children[i].name;
                f_children>>children[i].pol;
                f_children>>children[i].rost;
        }
        int col=0;
        int sum=0;
        for (i=0;i<N;i++)
        {
                if (children[i].pol=="m")
                { col=col+1;
                sum=sum+children[i].rost;
                }
        }
        avg_rost=sum/col;
```

```
        cout<<"srednii rost malchicov sostavljet"<<avg_rost;
        getch();
}
```

Варианты заданий

Здание 1

1. Дан файл вещественных чисел a.txt Найти количество отрицательных и количество положительных элементов.
2. Дан файл вещественных чисел a.txt Найти *количеств*о нулевых элементов и произведение элементов меньших 1 и больших 0.
3. Дан файл вещественных чисел a.txt Найти количество нулевых элементов и сумму отрицательных элементов.
4. Дан файл вещественных чисел a.txt Найти количество элементов равных 5 и сумму положительных элементов.
5. Дан файл вещественных чисел a.txt Найти количество нулевых элементов и сумму положительных элементов.
6. Дан файл вещественных чисел a.txt Найти количество положительных элементов и сумму положительных элементов.
7. Дан файл вещественных чисел a.txt Найти количество отрицательных и количество положительных элементов.
8. Дан файл вещественных чисел a.txt Найти *количеств*о нулевых элементов и произведение элементов меньших 1 и больших 0.
9. Дан файл вещественных чисел a.txt Найти количество нулевых элементов и сумму отрицательных элементов.
10. Дан файл вещественных чисел a.txt Найти количество элементов равных 5 и сумму положительных элементов.
11. Дан файл вещественных чисел a.txt Найти количество нулевых элементов и сумму положительных элементов.
12. Дан файл вещественных чисел a.txt Найти количество положительных элементов и сумму положительных элементов.
13. Дан файл вещественных чисел a.txt Найти *количеств*о положительных элементов и произведение элементов меньших 1 и больших 0.
14. Дан файл вещественных чисел a.txt Переписать положительные элементы в файл b.txt

Здание 2

1. Дано 2 файла вещественных чисел a1.txt и a2.txt. Найти сумму положительных элементов в двух файлов.
2. Дано 2 файла вещественных чисел a1.txt и a2.txt. Найти произведение отрицательных элементов двух файлов
3. Дано 2 файла вещественных чисел a1.txt и a2.txt. Найти количество нулевых элементов в двух файлов
4. Дано 2 файла вещественных чисел a1.txt и a2.txt. Найти сумму положительных элементов в двух файлов.
5. Дано 2 файла вещественных чисел a1.txt и a2.txt. Найти произведение отрицательных элементов двух файлов

6. Дано 2 файла вещественных чисел a1.txt и a2.txt. Найти количество нулевых элементов в двух файлов

7. Дан файл вещественных чисел a.txt . Переписать в файл a2.txt положительные элементы файла b(n) умноженные на 5.

8. Дан файл вещественных чисел a.txt . Переписать в файл a2.txt все ненулевые элементы файла a.txt

9. Дан файл вещественных чисел a.txt или a.txt. Переписать в файл a2.txt ненулевые элементы файла a.txt разделенные на 5.

10. Дан файл вещественных чисел a.txt Переписать в файл a2.txt отрицательные элементы файла a.txt умноженные на 2.

11. Дано 2 файла вещественных чисел a1.txt и a2.txt. В каком из двух данных файлов больше отрицательных элементов?

12. В данном файле a2.txt каких элементов больше, равных 0 или равных 1?

13. Дано 2 файла вещественных чисел a1.txt и a2.txt. Найти произведение отрицательных элементов двух файлов

14. Дано 2 файла вещественных чисел a1.txt и a2.txt. Найти произведение отрицательных элементов двух файлов

Задание №3

1. Организовать текстовый файл. Заменить в файле все маленькие латинские буквы на большие.(создавая новый дополнительный файл)

2. Из заданного входного файла считать символы и записать в один новый файл только буквы, в другой новый файл только цифры .

3. Организовать текстовый файл. Организовать замену символов в файле. "старый" символ и "новый" символ запрашиваются и вводятся с клавиатуры.(создавая новый дополнительный файл)

4. Организовать текстовый файл. Преобразовать файл, удалив в нем лишние пробелы.(создавая новый дополнительный файл)

5. Организовать текстовый файл, состоящий из строк. Заменить в файле все большие латинские буквы на маленькие . создавая новый дополнительный файл)

6. Организовать текстовый файл. Заменить в файле все цифры на '*'.(создавая новый дополнительный файл)

7. Организовать текстовый файл. Заменить в файле все буквы (нецифры) на '*'.(создавая новый дополнительный файл)

8. Организовать текстовый файл. Удалить в файле все цифры. (создавая новый дополнительный файл)

9. Из заданного входного файла считать символы и записать в один новый файл только большие латинские буквы, в другой новый файл только малые латинские буквы и посчитать количество цифр.

10. Организовать текстовый файл. Удалить в файле все буквы. (создавая новый дополнительный файл)

11. Из заданного входного файла считать символы и записать в новый файл все символы за исключением символов разделителей : пробелы , точки , запятые, двоеточия и т.д.

12. Организовать текстовый файл . Оставив в файле только буквы. (создавая новый дополнительный файл)

13. Из заданного входного файла считать символы и записать в новый файл только большие буквы латинского алфавита .

14. Организовать файл вещественных чисел Заменить все положительные компоненты файла их квадратными корнями, а все отрицательные компоненты их квадратами. . создавая новый дополнительный файл)

15. Организовать файл целых чисел . Удалить из файла все отрицательные компоненты. . создавая новый дополнительный файл)

16. организовать файл целых чисел, заменить все элементы файла от -10 до 10 на противоположные. создавая новый дополнительный файл)

17. Организовать файл целых чисел Все числа, кратные 3 заменить их удвоенным произведением. создавая новый дополнительный файл)

18. организовать файл целых чисел, заменить все элементы файла от -2 до 5 на противоположные. создавая новый дополнительный файл)

Здание 4.

1. Организовать файл целых чисел. В новый файл записать элементы файла занимающие нечётные позиции, в другой новый файл элементы файла занимающие чётные позиции.

2. Организовать файл целых чисел. Определить наибольший отрицательный компонент файла среди компонент файла расположенных на чётных позициях.

3. Организовать файл целых чисел . Определить наибольший элемент файла среди элементов файла номера которых кратны трем.

4. Организовать файл целых чисел. Вычислить количество отрицательных компонентов файла расположенных на нечётных позициях.

5. Организовать файл целых чисел . Определить наименьший положительный компонент файла среди компонент файла расположенных на чётных позициях.

6. Организовать файл целых чисел. Вычислить среднее значение среди положительных значений файла, номера которых кратны трем.

7. Организовать файл целых чисел . Определить наименьший положительный компонент файла среди компонент файла расположенных на нечётных позициях.

8. Организовать файл целых чисел. Вычислить количество отрицательных компонентов файла расположенных на чётных позициях.

9. Организовать файл целых чисел. Вычислить среднее значение среди положительных значений файла расположенных на чётных позициях.

10. Организовать файл целых чисел. Вычислить количество отрицательных компонентов файла , номера которых кратны трем.

11. Организовать файл целых чисел. Найти сумму элементов файла расположенных на чётных позициях.

12. Организовать файл целых чисел. Вычислить среднее значение среди положительных значений файла расположенных на нечётных позициях.

13. Организовать файл целых чисел. Найти сумму элементов файла , номера которых кратны трем.

14. Организовать файл целых чисел. Вычислить количество нулевых элементов файла расположенных на чётных позициях.

15. Организовать файл целых чисел . Определить наибольший отрицательный компонент файла среди компонент файла расположенных на нечётных позициях.

16. Организовать файл целых чисел. Найти сумму элементов файла расположенных на нечётных позициях.

17. Организовать файл целых чисел. Вычислить количество нулевых элементов файла расположенных на нечётных позициях.

ЛАБОРАТОРНАЯ РАБОТА №13. ОБРАБОТКА ТЕКСТОВЫХ ФАЙЛОВ

Цель лабораторной работы

Разработка программ языке С++ для обработки текстовых файлов .

Методические указания

С++. Работа с текстовыми файлами

Файлы позволяют пользователю считывать большие объемы данных непосредственно с диска, не вводя их с клавиатуры. Существуют два основных типа файлов: **текстовые и двоичные**.

Текстовыми называются файлы, состоящие из любых символов. Они организуются по строкам, каждая из которых заканчивается символом «*конца строки*». Конец самого файла обозначается символом «*конца файла*». При записи информации в текстовый файл, просмотреть который можно с помощью любого текстового редактора, все данные преобразуются к символьному типу и хранятся в символьном виде.

В *двоичных* файлах информация считывается и записывается в виде блоков определенного размера, в которых могут храниться данные любого вида и структуры.

Для работы с файлами используются специальные типы данных, называемые *потоками.* Поток **ifstream** служит для работы с файлами в режиме чтения, а **ofstream** в режиме записи. Для работы с файлами в режиме как записи, так и чтения служит поток **fstream.**

В программах на С++ при работе с текстовыми файлами необходимо подключать библиотеки **iostream** и **fstream.**

Для того чтобы записывать данные в текстовый файл, необходимо:

1. описать переменную типа **ofstream.**
2. открыть файл с помощью функции **open.**
3. вывести информацию в файл.
4. обязательно закрыть файл.

Для считывания данных из текстового файла, необходимо:

1. описать переменную типа **ifstream.**
2. открыть файл с помощью функции **open.**
3. считать информацию из файла, при считывании каждой порции данных необходимо проверять, достигнут ли конец файла.
4. закрыть файл.

Запись информации в текстовый файл

Как было сказано ранее, для того чтобы начать работать с текстовым файлом, необходимо описать переменную типа **ofstream**. Например, так:

ofstream F;

Будет создана переменная **F** для записи информации в файл. На следующим этапе файл необходимо открыть для записи. В общем случае оператор открытия потока будет иметь вид:

F.open(«file», mode);

Здесь **F** — переменная, описанная как **ofstream**, **file** — полное имя файла на диске, **mode** — режим работы с открываемым файлом. Обратите внимание на то, что при указании полного имени файла нужно ставить двойной слеш. Для обращения, например к файлу **accounts.txt,** находящемуся в папке **sites** на диске **D**, в программе необходимо указать: **D:\\sites\\accounts.txt.**

Файл может быть открыт в одном из следующих режимов:

- ***ios::in*** — открыть файл в режиме чтения данных; режим является режимом по умолчанию для потоков **ifstream**;
- ***ios::out*** — открыть файл в режиме записи данных (при этом информация о существующем файле уничтожается); режим является режимом по умолчанию для потоков **ofstream**;
- ***ios::app*** — открыть файл в режиме записи данных в конец файла;
- ***ios::ate*** — передвинуться в конец уже открытого файла;
- ***ios::trunc*** — очистить файл, это же происходит в режиме ios::out;
- ***ios::nocreate*** — не выполнять операцию открытия файла, если он не существует;
- ***ios::noreplace*** — не открывать существующий файл.

Параметр mode может отсутствовать, в этом случае файл открывается в режиме по умолчанию для данного потока.

После удачного открытия файла (в любом режиме) в переменной **F** будет храниться **true**, в противном случае **false**. Это позволит проверить корректность операции открытия файла.

Открыть файл (в качестве примера возьмем файл **D:\\sites\\accounts.txt**) в режиме записи можно одним из следующих способов:

```
    //первый способ
ofstream F;
F.open("D:\\sites\<strong>\</strong>accounts<strong>.</strong>txt", ios::out);
//второй способ, режим ios::out является режимом по умолчанию

    //для потока ofstream
ofstream F;
F.open("D:\\game\\noobs.txt");
//третий способ объединяет описание переменной и типа поток
//и открытие файла в одном операторе
ofstream F ("D:\\game\\noobs.txt", ios::out);
```

После открытия файла в режиме записи будет создан пустой файл, в который можно будет записывать информацию.

Если вы хотите открыть существующий файл в режиме дозаписи, то в качестве режима следует использовать значение **ios::app**.

После открытия файла в режиме записи, в него можно писать точно так же, как и на экран, только вместо стандартного устройства вывода **cout** необходимо указать имя открытого файла.

Например, для записи в поток **F** переменной **a**, оператор вывода будет иметь вид:

F<<a;

Для последовательного вывода в поток **G** переменных **b**, **c**, **d** оператор вывода станет таким:

G<<b<<c<<d;

Закрытие потока осуществляется с помощью оператора:

F.close();

В качестве примера рассмотрим следующую задачу.

Задача 1

Создать текстовый файл **D:\\sites\\accounts.txt** и записать в него **n** вещественных чисел.

Решение

```
	#include "stdafx.h"
#include <iostream>
#include <fstream>
#include <iomanip>
using namespace std;
int main()
{
setlocale (LC_ALL, "RUS");
int i, n;
double a;
ofstream f;                    //описывает поток для записи данных в файл
//открываем файл в режиме записи,
//режим ios::out устанавливается по умолчанию
f.open("D:\\sites\\accounts.txt", ios::out);
cout<<"n="; cin>>n;                //вводим количество вещественных чисел
for (i=0; i<n; i++)            //цикл для ввода вещественных чисел и записи их в файл
	{
cout<<"a=";
cin>>a; //ввод числа
f<<a<<"\t";
}
f.close();                     //закрытие потока
system("pause");
return 0;
}
```

Чтение информации из текстового файла

Для того чтобы прочитать информацию из текстового файла, необходимо описать переменную типа **ifstream**. После этого нужно открыть файл для чтения с помощью оператора **open**. Если переменную назвать **F**, то первые два оператора будут такими:

```
	ifstream F;
F.open("D:\\sites\\accounts.txt", ios::in);
```

После открытия файла в режиме чтения из него можно считывать информацию точно так же, как и с клавиатуры, только вместо **cin** нужно указать имя потока, из которого будет происходить чтение данных.

Например, для чтения данных из потока **F** в переменную **a**, оператор ввода будет выглядеть так:

F>>a;

Два числа в текстовом редакторе считаются разделенными, если между ними есть хотя бы один из символов: пробел, табуляция, символ конца строки. Хорошо, когда программисту заранее известно, сколько и какие значения хранятся в текстовом файле. Однако часто известен лишь тип значений, хранящихся в файле, при этом их количество может быть различным. Для решения данной проблемы необходимо считывать значения из файла поочередно, а перед каждым считыванием проверять, достигнут ли конец

файла. А поможет сделать это функция **F.eof()**. Здесь **F -** имя потока функция возвращает логическое значение: **true** или **false**, в зависимости от того достигнут ли конец файла.

Следовательно, цикл для чтения содержимого всего файла можно записать так:

```
    //организуем для чтения значений из файла, выполнение
//цикла прервется, когда достигнем конец файла,
//в этом случае F.eof() вернет истину
while (!F.eof())
{
   F>>a; //чтение очередного значения из потока F в переменную a
  //далее идет обработка значения переменной a
}
```

Для лучшего усвоения материала рассмотрим задачу.

Задача 2

В текстовом файле D:\\game\\accounts.txt хранятся вещественные числа, вывести их на экран и вычислить их количество.

Решение

```
    #include "stdafx.h"
#include <iostream>
#include <fstream>
#include <iomanip>
#include <stdlib.h>
using namespace std;
int main()
{
setlocale (LC_ALL, "RUS");
int n=0;
float a;
fstream F;
F.open("D:\\sites\\accounts.txt");          //открываем файл в режиме чтения

if (F)                          //если открытие файла прошло корректно, то
{
//цикл для чтения значений из файла; выполнение цикла прервется,
//когда достигнем конца файла, в этом случае F.eof() вернет истину.
while (!F.eof())
{
   F>>a;              //чтение очередного значения из потока F в переменную a
  cout<<a<<"\t";   //вывод значения переменной a на экран
   n++;  //увеличение количества считанных чисел
}
F.close();//закрытие потока
cout<<"n="<<n<<endl;          //вовод на экран количества считанных чисел
}
//если открытие файла прошло некорректно, то вывод
//сообщения об отсутствии такого файла
```

```
else cout<<" Файл не существует"<<endl;
system("pause");
return 0;
}
```

На этом относительно объемный урок по текстовым файлам закончен. В следующей статье будут рассмотрены методы манипуляции, при помощи которых в C++ обрабатываются двоичные файлы.

Варианты заданий

1. Сформировать массив, каждый элемент которого имеет следующую структуру: студент=ФИО; пол:(м,ж); рост

Вывести на экран сообщение пользователю вида:

Выберете номер действия:

1--загрузить все данные из файла на экран;

2--ввести новые данные в файл ;

3-- добавить данных в конец файла;

3--сортировка базы данных по алфавиту (по фамилии);

4--сортировка базы данных по росту;

5--вычисление : средний рост женщин;

6—вывод: имя самого высокого мужчины;

0--выход из программы;

Ваш выбор___

Основную программу организовать в виде цикла do – while «пока не выбран пункт 0—выход».

Для вариантов выбора использовать оператор SWITCH - CASE.

Каждый пункт меню (действие) оформить виде метода (подпрограммы-функции или подпрограммы-процедуры)

*и по желанию разместить их в отдельном модуле .

** (дополнительно) При выборе пунктов №1 №2№3 запросить имя и путь к файлу для отображения (чтения) базы и осуществить

***проверку возможности открытия данного файла.

2. Сформировать массив, каждый элемент которого имеет следующую структуру

служащий=ФИО: string; число рождения; месяц рождения ; год рождения ; пол:(м,ж);

Вывести на экран сообщение пользователю вида:

Выберете номер действия:

1--загрузить все данные из файла на экран;

2--ввести новые данные в файл ;

3-- добавить данных в конец файла;

3--сортировка базы данных по алфавиту (по фамилии);

4--сортировка базы данных по возрасту;

5--вывод : список людей, родившихся в заданном месяце*;*

6—вывод: фамилию самого старшего мужчины*;*

7-- вывод: все фамилии, начинающиеся с заданной буквы;

0--выход из программы;

Ваш выбор___

Основную программу организовать в виде цикла do – while «пока не выбран пункт 0—выход».

Для вариантов выбора использовать оператор SWITCH - CASE.

Каждый пункт меню (действие) оформить виде метода (подпрограммы-функции или подпрограммы-процедуры)

*и по желанию разместить их в отдельном модуле .

** (дополнительно) При выборе пунктов №1 №2№3 запросить имя и путь к файлу для отображения (чтения) базы и осуществить

***проверку возможности открытия данного файла.

3. Сформировать массив, каждый элемент которого имеет следующую структуру знакомый= фамилия: string; номертел:10000..99999; адрес:string;

Вывести на экран сообщение пользователю вида:

Выберете номер действия:

1. *загрузить все данные из файла на экран;*
2. *ввести новые данные в файл ;*
3. *добавить данных в конец файла;*
4. *сортировка базы данных по алфавиту (по фамилии);*
5. *есть ли в книжке телефон данного человека;*
6. *определить кому принадлежит данный телефон;*
7. *вывод: список людей, живущих на данной улице.;*

0--выход из программы;

Ваш выбор___

Основную программу организовать в виде цикла do – while «пока не выбран пункт 0—выход».

Для вариантов выбора использовать оператор SWITCH - CASE.

Каждый пункт меню (действие) оформить виде метода (подпрограммы-функции или подпрограммы-процедуры)

*и по желанию разместить их в отдельном модуле .

** (дополнительно) При выборе пунктов №1 №2№3 запросить имя и путь к файлу для отображения (чтения) базы и осуществить

***проверку возможности открытия данного файла.

4Сформировать массив, каждый элемент которого имеет следующую структуру студент= фамилия: string; номергр: string; оценка1: integer; оценка2: integer; оценка3: integer;

Вывести на экран сообщение пользователю вида:

Выберете номер действия:

1. *загрузить все данные из файла на экран;*
2. *ввести новые данные в файл*
3. *добавить данных в конец файла;*
4. *сортировка базы данных по алфавиту (по фамилии);*
5. список задолжников*;*
6. *фамилию того, кто лучше всех сдал экзамены

7. *определить* средний балл по данному предмету

0--выход из программы;

Ваш выбор___

Основную программу организовать в виде цикла do – while «пока не выбран пункт 0—выход».

Для вариантов выбора использовать оператор SWITCH - CASE.

Каждый пункт меню (действие) оформить виде метода (подпрограммы-функции или подпрограммы-процедуры)

*и по желанию разместить их в отдельном модуле .

** (дополнительно) При выборе пунктов №1 №2№3 запросить имя и путь к файлу для отображения (чтения) базы и осуществить

***проверку возможности открытия данного файла.

5. Сформировать массив, каждый элемент которого имеет следующую структуру

студент= фамилия:string; имя:string; пол:(м,ж); возраст:16..35; курс:1..5;

Вывести на экран сообщение пользователю вида:

Выберете номер действия:

1. *загрузить все данные из файла на экран;*

2. *ввести новые данные в файл ;*

3. *добавить данных в конец файла;*

4. *сортировка базы данных по алфавиту (по фамилии);*

5. *сортировка базы данных по возрасту;*

6. ***определить: курс, на котором наибольший процент мужчин;*

7. *список студентов данного пола, данного курса*

0--выход из программы;

Ваш выбор___

Основную программу организовать в виде цикла do – while «пока не выбран пункт 0—выход».

Для вариантов выбора использовать оператор SWITCH - CASE.

Каждый пункт меню (действие) оформить виде метода (подпрограммы-функции или подпрограммы-процедуры)

*и по желанию разместить их в отдельном модуле .

** (дополнительно) При выборе пунктов №1 №2№3 запросить имя и путь к файлу для отображения (чтения) базы и осуществить

***проверку возможности открытия данного файла.

6. Сформировать массив, каждый элемент которого имеет следующую структуру

пассажир= фамилия:string; имя:string; номер рейса:string; количество вещей:integer; общий вес: integer;

Вывести на экран сообщение пользователю вида:

Выберете номер действия:

1. *загрузить все данные из файла на экран;*

2. *ввести новые данные в файл ;*

3. *добавить данных в конец файла;*

4. *сортировка базы данных по алфавиту (по фамилии);*

5. *определить: пассажира с наибольшим количеством вещей;*

6. вывести список пассажиров и информацию об их багаже, улетающих данным рейсом (номер рейса вводит пользователь).

0--выход из программы;

Ваш выбор___

Основную программу организовать в виде цикла do – while «пока не выбран пункт 0—выход».

Для вариантов выбора использовать оператор SWITCH - CASE.

Каждый пункт меню (действие) оформить виде метода (подпрограммы-функции или подпрограммы-процедуры)

*и по желанию разместить их в отдельном модуле .

** (дополнительно) При выборе пунктов №1 №2№3 запросить имя и путь к файлу для отображения (чтения) базы и осуществить

***проверку возможности открытия данного файла.

7. Сформировать массив, каждый элемент которого имеет следующую структуру

владелец= фамилия:string; адрес:string; марка автомобиля:string; рег. номер:string; год выпуска: ;

Вывести на экран сообщение пользователю вида:

Выберете номер действия:

1. загрузить все данные из файла на экран;

2. ввести новые данные в файл ;

3. добавить данных в конец файла;

4. сортировка базы данных по алфавиту (по фамилии);

5. сортировка базы данных по году выпуска

6. определить : владельца самого старого автомобиля;

7. вывести фамилии владельцев и номера автомобилей данной марки (введенной пользователем)

0--выход из программы;

Ваш выбор___

Основную программу организовать в виде цикла do – while «пока не выбран пункт 0—выход».

Для вариантов выбора использовать оператор SWITCH - CASE.

Каждый пункт меню (действие) оформить виде метода (подпрограммы-функции или подпрограммы-процедуры)

*и по желанию разместить их в отдельном модуле .

** (дополнительно) При выборе пунктов №1 №2№3 запросить имя и путь к файлу для отображения (чтения) базы и осуществить

***проверку возможности открытия данного файла.

8. Сформировать массив, каждый элемент которого имеет следующую структуру

ребенок= фамилия:string; адрес:string; пол:(муж,жен); количество дней посещения:integer;

Вывести на экран сообщение пользователю вида:

Выберете номер действия:

1. загрузить все данные из файла на экран;

2. ввести новые данные в файл ;

3. добавить данных в конец файла;

4. сортировка базы данных по алфавиту (по фамилии);

5. определить : самого болеющего ребенка;

6. вывести список детей проживающих на данной улице (введенной пользователем)

7. определить: кто больше болеет мальчики или девочки

0--выход из программы;

Ваш выбор___

Основную программу организовать в виде цикла do – while «пока не выбран пункт 0—выход».

Для вариантов выбора использовать оператор SWITCH - CASE.

Каждый пункт меню (действие) оформить виде метода (подпрограммы-функции или подпрограммы-процедуры)

*и по желанию разместить их в отдельном модуле .

** (дополнительно) При выборе пунктов №1 №2№3 запросить имя и путь к файлу для отображения (чтения) базы и осуществить

***проверку возможности открытия данного файла.

9. Сформировать массив, каждый элемент которого имеет следующую структуру

книга= автор:string; название:string; год издание:integer; издательство:string; количество страниц:integer;

Вывести на экран сообщение пользователю вида:

Выберете номер действия:

1. загрузить все данные из файла на экран;

2. ввести новые данные в файл ;

3. добавить данных в конец файла;

4. сортировка базы данных по алфавиту (по названию)

5. определить:есть ли в библиотеке книги данного автора;

6. найти книгу с наибольшим количеством страниц

7. найти названия книг данного автора, изданных с указанного года,

0--выход из программы;

Ваш выбор___

Основную программу организовать в виде цикла do – while «пока не выбран пункт 0—выход».

Для вариантов выбора использовать оператор SWITCH - CASE.

Каждый пункт меню (действие) оформить виде метода (подпрограммы-функции или подпрограммы-процедуры)

*и по желанию разместить их в отдельном модуле .

** (дополнительно) При выборе пунктов №1 №2№3 запросить имя и путь к файлу для отображения (чтения) базы и осуществить

***проверку возможности открытия данного файла.

10. Сформировать массив, каждый элемент которого имеет следующую структуру

товар= наименование:string; страна:string; объем партии:integer; цена:integer;

Вывести на экран сообщение пользователю вида:

Выберете номер действия:

1. загрузить все данные из файла на экран;

2. ввести новые данные в файл ;

3. добавить данных в конец файла;

4. сортировка базы данных по алфавиту (по названию)

*5. *определить: страну, в которую экспортируется товар на максимальную сумму;*

6. вывести список стран, в которые экспортируется данный товар (введенный пользователем)

7. найти товары, который имеет минимальный объем партии.

0--выход из программы;

Ваш выбор___

Основную программу организовать в виде цикла do – while «пока не выбран пункт 0—выход».

Для вариантов выбора использовать оператор SWITCH - CASE.

Каждый пункт меню (действие) оформить виде метода (подпрограммы-функции или подпрограммы-процедуры)

*и по желанию разместить их в отдельном модуле .

** (дополнительно) При выборе пунктов №1 №2№3 запросить имя и путь к файлу для отображения (чтения) базы и осуществить

***проверку возможности открытия данного файла.

11. Сформировать массив, каждый элемент которого имеет следующую структуру

игрушка=название:string; цена:integer; возраст_от; возраст_до;

Вывести на экран сообщение пользователю вида:

Выберете номер действия:

1. загрузить все данные из файла на экран;

2. ввести новые данные в файл ;

3. добавить данных в конец файла;

4. сортировка базы данных по алфавиту (по названию)

5. определить: название самой дорогой игрушки

6. вывести список игрушек, которые подходят детям данного возраста (введенный пользователем)

*7. *подобрать игрушки на данную сумму денег (все варианты).*

8. 0--выход из программы;

Ваш выбор___

Основную программу организовать в виде цикла do – while «пока не выбран пункт 0—выход».

Для вариантов выбора использовать оператор SWITCH - CASE.

Каждый пункт меню (действие) оформить виде метода (подпрограммы-функции или подпрограммы-процедуры)

*и по желанию разместить их в отдельном модуле .

** (дополнительно) При выборе пунктов №1 №2№3 запросить имя и путь к файлу для отображения (чтения) базы и осуществить

***проверку возможности открытия данного файла.

12. Сформировать массив, каждый элемент которого имеет следующую структуру

игрушка= название:string; цена:integer; цвет:string; возраст_от; возраст_до;

Вывести на экран сообщение пользователю вида:

Выберете номер действия:

1. загрузить все данные из файла на экран;

2. ввести новые данные в файл ;

3. добавить данных в конец файла;

4. сортировка базы данных по алфавиту (по названию)

5. определить: название игрушек, цена которых не превышает данную и которые подходят детям данного возраста;

6. найти самую дешевую игрушку данного названия;

*7. * найти самый распространенный цвет игрушек.*

8. 0--выход из программы;

Ваш выбор___

Основную программу организовать в виде цикла do – while «пока не выбран пункт 0—выход».

Для вариантов выбора использовать оператор SWITCH - CASE.

Каждый пункт меню (действие) оформить виде метода (подпрограммы-функции или подпрограммы-процедуры)

*и по желанию разместить их в отдельном модуле .

** (дополнительно) При выборе пунктов №1 №2№3 запросить имя и путь к файлу для отображения (чтения) базы и осуществить

***проверку возможности открытия данного файла.

13. Сформировать массив, каждый элемент которого имеет следующую структуру

пассажир= фамилия:string; имя:string; номер рейса:string; количество вещей:integer; общий вес:integer;

Вывести на экран сообщение пользователю вида:

Выберете номер действия:

1. загрузить все данные из файла на экран;

2. ввести новые данные в файл ;

3. добавить данных в конец файла;

4. сортировка базы данных по алфавиту (по номерам рейсов)

5. определить: число пассажиров, количество вещей которых превосходит среднее число вещей;

6. найти пассажира с данным количеством вещей и не более данного веса;(колич и вес вводит пользователь)

*7. *вывести информацию о количестве вещей и общем весе каждого рейса.*

0--выход из программы;

Ваш выбор___

Основную программу организовать в виде цикла do – while «пока не выбран пункт 0—выход».

Для вариантов выбора использовать оператор SWITCH - CASE.

Каждый пункт меню (действие) оформить виде метода (подпрограммы-функции или подпрограммы-процедуры)

*и по желанию разместить их в отдельном модуле .

** (дополнительно) При выборе пунктов №1 №2№3 запросить имя и путь к файлу для отображения (чтения) базы и осуществить

***проверку возможности открытия данного файла.

14. Сформировать массив, каждый элемент которого имеет следующую структуру

спортсмен= фамилия:string; страна:string; рост: ; вес: ; год рождения: int; результат (номер места в рейтинге) :int ;

Вывести на экран сообщение пользователю вида:

Выберете номер действия:

1. загрузить все данные из файла на экран;

2. ввести новые данные в файл ;

3. добавить данных в конец файла;

4. сортировка базы данных по алфавиту (по фамилии)

5. определить: средний рост и вес спортсменов данной страны;

6. найти лучшего спортсмена данной страны;*(страну вводит пользователь)*

7. список спортсменов данного возраста с результатом, не хуже данного; *(исходные вводит пользователь)*

0--выход из программы;

Ваш выбор___

Основную программу организовать в виде цикла do – while «пока не выбран пункт 0—выход».

Для вариантов выбора использовать оператор SWITCH - CASE.

Каждый пункт меню (действие) оформить виде метода (подпрограммы-функции или подпрограммы-процедуры)

*и по желанию разместить их в отдельном модуле .

** (дополнительно) При выборе пунктов №1 №2№3 запросить имя и путь к файлу для отображения (чтения) базы и осуществить

***проверку возможности открытия данного файла.

15. Сформировать массив, каждый элемент которого имеет следующую структуру

спортсмен= фамилия:string; страна:string; тренер:string; год рождения:int; результат(номер места в рейтинге):int ;

Вывести на экран сообщение пользователю вида:

Выберете номер действия:

1. загрузить все данные из файла на экран;

2. ввести новые данные в файл ;

3. добавить данных в конец файла;

4. сортировка базы данных по рейтингу

5. найти самого молодого спортсмена, занимающегося у данного тренера;

*6. *найти лучшего тренера данной страны; (страну вводит пользователь)*

*7. *список тренеров с указанием страны.*

0--выход из программы;

Ваш выбор___

Основную программу организовать в виде цикла do – while «пока не выбран пункт 0—выход».

Для вариантов выбора использовать оператор SWITCH - CASE.

Каждый пункт меню (действие) оформить виде метода (подпрограммы-функции или подпрограммы-процедуры)

*и по желанию разместить их в отдельном модуле .

** (дополнительно) При выборе пунктов №1 №2№3 запросить имя и путь к файлу для отображения (чтения) базы и осуществить

***проверку возможности открытия данного файла.

ЛАБОРАТОРНАЯ РАБОТА №14. ОДНОСВЯЗНЫЕ ДИНАМИЧЕСКИЕ СПИСКИ

Цель лабораторной работы

Разработка программ языке С++ с использованием односвязных динамических списков

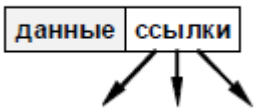
Методические указания

Динамические структуры данных

Часто в серьезных программах надо использовать данные, размер и структура которых должны меняться в процессе работы. Динамические массивы здесь не выручают, поскольку заранее нельзя сказать, сколько памяти надо выделить – это выясняется только в процессе работы. Например, надо проанализировать текст и определить, какие слова и в каком количество в нем встречаются, причем эти слова нужно расставить по алфавиту.

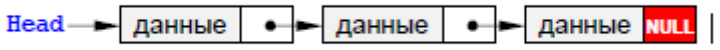
В таких случаях применяют данные особой структуры, которые представляют собой отдельные элементы, связанные с помощью **ссылок.**

Каждый элемент (**узел**) состоит из двух областей памяти: **поля данных** и **ссылок.** Ссылки – это адреса других узлов этого же типа, с которыми данный элемент логически связан. В языке Си для организации ссылок используются переменные-указатели. При добавлении нового узла в такую структуру выделяется новый блок памяти и (с помощью ссылок) устанавливаются связи этого элемента с уже существующими. Для обозначения конечного элемента в цепи используются **нулевые ссылки** (**NULL**).



Линейный список

В простейшем случае каждый узел содержит всего одну ссылку. Для определенности будем читать, что решается задача частотного анализа текста – определения всех слов, встречающихся в тексте и их количества. В этом случае область данных элемента включает строку (длиной е более 40 символов) и целое число.



Каждый элемент содержит также ссылку на **следующий** за ним элемент. У последнего в списке лемента поле ссылки содержит **NULL**. Чтобы не потерять список, мы должны где-то (в переменной) хранить адрес его первого узла – он называется «головой» списка. В программе надо бъявить два новых типа данных – узел списка **Node** и указатель на него **PNode**. Узел представляет собой структуру, которая содержит три поля - строку, целое число и указатель на такой же узел. Правилами языка Си допускается объявление

```
struct Node {
        char word[40]; // область данных
        int count;
        Node *next; // ссылка на следующий узел
};
typedef Node *PNode; // тип данных: указатель на узел
```

В дальнейшем мы будем считать, что указатель Head указывает на начало списка, то есть, объявлен в виде

PNode Head = NULL;

Первая буква «P» в названии типа PNode происходит от слова *pointer* – указатель (англ.) В начале работы в списке нет ни одного элемента, поэтому в указатель Head записывается нулевой адрес NULL.

Создание элемента списка

Для того, чтобы добавить узел к списку, необходимо создать его, то есть выделить память под узел и запомнить адрес выделенного блока. Будем считать, что надо добавить к списку узел, соответствующий новому слову, которое записано в переменной NewWord. Составим функцию, которая создает новый узел в памяти и возвращает его адрес. Обратите внимание, что при записи данных в узел используется обращение к полям структуры через указатель.

```
PNode CreateNode ( char NewWord[] )
{
        PNode NewNode = new Node; // указатель на новый узел
        strcpy(NewNode->word, NewWord); // записать слово
        NewNode->count = 1; // счетчик слов = 1
        NewNode->next = NULL; // следующего узла нет
        return NewNode; // результат функции – адрес узла
}
```

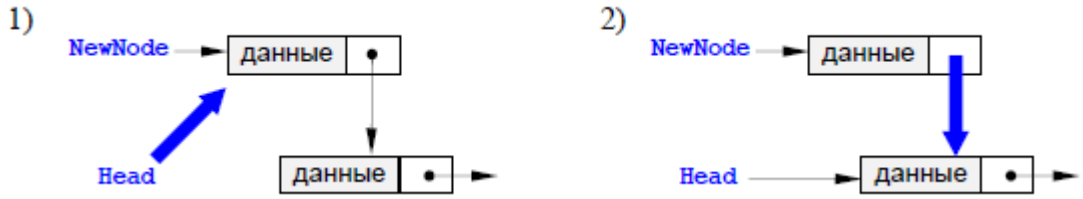
После этого узел надо добавить к списку (в начало, в конец или в середину).

Добавление узла

Добавление узла в начало списка

При добавлении нового узла NewNode в начало списка надо 1) установить ссылку узла

NewNode на голову существующего списка и 2) установить голову списка на новый узел.



По такой схеме работает процедура AddFirst. Предполагается, что адрес начала списка хранится в Head. Важно, что здесь и далее адрес начала списка передается *по ссылке*, так как придобавлении нового узла он изменяется внутри процедуры.

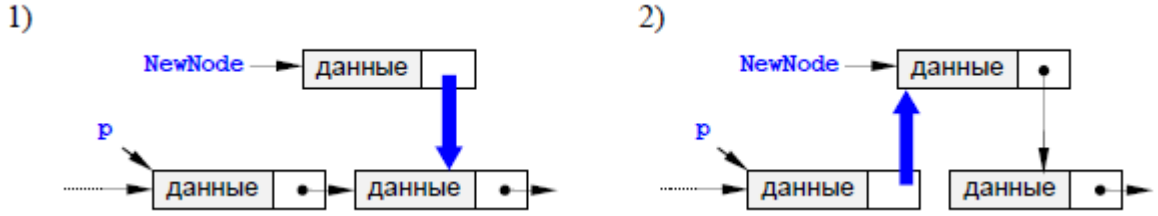
```
void AddFirst (PNode &Head, PNode NewNode)
{
        NewNode->next = Head;
        Head = NewNode;
}
```

Добавление узла после заданного

Дан адрес NewNode нового узла и адрес p одного из существующих узлов в списке. Требуется вставить в список новый узел после узла с адресом p. Эта операция выполняется в два этапа:

1) установить ссылку нового узла на узел, следующий за данным;

2) установить ссылку данного узла p на NewNode.



Последовательность операций менять нельзя, потому что если сначала поменять ссылку у узла p, будет потерян адрес следующего узла.

```
void AddAfter (PNode p, PNode NewNode)
{
        NewNode->next = p->next;
        p->next = NewNode;
}
```

Добавление узла перед заданным

Эта схема добавления самая сложная. Проблема заключается в том, что в простейшем линейном списке (он называется *односвязным*, потому что связи направлены только в одну сторону) для того, чтобы получить адрес предыдущего узла, нужно пройти весь список сначала.

Задача сведется либо к вставке узла в начало списка (если заданный узел – первый), либо к вставке после заданного узла.

```
void AddBefore(PNode &Head, PNode p, PNode NewNode)
{
        PNode q = Head;
        if (Head == p) {
        AddFirst(Head, NewNode); // вставка перед первым узлом
        return;
}
while (q && q->next!=p) // ищем узел, за которым следует p
        q = q->next;
        if ( q ) // если нашли такой узел,
        AddAfter(q, NewNode); // добавить новый после него
}
```

Такая процедура обеспечивает «защиту от дурака»: если задан узел, не присутствующий в списке, то в конце цикла указатель q равен NULL и ничего не происходит.

Существует еще один интересный прием: если надо вставить новый узел NewNode до заданного узла p, вставляют узел после этого узла, а потом выполняется обмен данными между узлами NewNode и p. Таким образом, по адресу p в самом деле будет расположен узел с новыми данными, а по адресу NewNode – с теми данными, которые были в узле p, то есть мы решили задачу. Этот прием не сработает, если адрес

нового узла NewNode запоминается где-то в программе и потом используется, поскольку по этому адресу будут находиться другие данные.

 Добавление узла в конец списка

Для решения задачи надо сначала найти последний узел, у которого ссылка равна NULL, а затем воспользоваться процедурой вставки после заданного узла. Отдельно надо обработать случай, когда список пуст.

```
void AddLast(PNode &Head, PNode NewNode)
{
        PNode q = Head;
        if (Head == NULL) { // если список пуст,
        AddFirst(Head, NewNode); // вставляем первый элемент
        return;
}
while (q->next) q = q->next; // ищем последний элемент
AddAfter(q, NewNode);
}
```

Проход по списку

Для того, чтобы пройти весь список и сделать что-либо с каждым его элементом, надо начать с головы и, используя указатель next, продвигаться к следующему узлу.

```
PNode p = Head; // начали с головы списка
while ( p != NULL ) { // пока не дошли до конца
        // делаем что-нибудь с узлом p
        p = p->next; // переходим к следующему узлу
}
```

Поиск узла в списке

Часто требуется найти в списке нужный элемент (его адрес или данные). Надо учесть, что требуемого элемента может и не быть, тогда просмотр заканчивается при достижении конца списка. Такой подход приводит к следующему алгоритму:

1) начать с головы списка;

2) пока текущий элемент существует (указатель – не NULL), проверить нужное условие и перейти к следующему элементу;

3) закончить, когда найден требуемый элемент или все элементы списка просмотрены.

Например, следующая функция ищет в списке элемент, соответствующий заданному слову (для которого поле word совпадает с заданной строкой NewWord), и возвращает его адрес или NULL, если такого узла нет.

```
PNode Find (PNode Head, char NewWord[])
{
        PNode q = Head;
        while (q && strcmp(q->word, NewWord))
        q = q->next;
        return q;
}
```

Вернемся к задаче построения алфавитно-частотного словаря. Для того, чтобы добавить новое слово в нужное место (в алфавитном порядке), требуется найти адрес узла, *перед* которым надо вставить новое слово. Это будет первый от начала списка

узел, для которого «его» слово окажется «больше», чем новое слово. Поэтому достаточно просто изменить условие в цикле while в функции Find., учитывая, что функция strcmp возвращает «разность» первого и второго слова.

```
PNode FindPlace (PNode Head, char NewWord[])
{
        PNode q = Head;
        while (q && (strcmp(q->word, NewWord) > 0))
        q = q->next;
        return q;
}
```

Эта функция вернет адрес узла, перед которым надо вставить новое слово (когда функция strcmp вернет положительное значение), или NULL, если слово надо добавить в конец списка.

Алфавитно-частотный словарь

Теперь можно полностью написать программу, которая обрабатывает файл input.txt и составляет для него алфавитно-частотный словарь в файле output.txt.

```
void main()
{
        PNode Head = NULL, p, where;
        FILE *in, *out;
        char word[80];
        int n;
        in = fopen ( "input.dat", "r" );
        while ( 1 ) {
                n = fscanf ( in, "%s", word ); // читаем слово из файла
                if ( n <= 0 ) break;
                p = Find ( Head, word ); // ищем слово в списке
                if ( p != NULL ) // если нашли слово,
                p->count ++; // увеличить счетчик
                else {
                p = CreateNode ( word ); // создаем новый узел
                where = FindPlace ( Head, word ); // ищем место
                if ( ! where )
                AddLast ( Head, p );
                else AddBefore ( Head, where, p );
                }
        }
        fclose(in);
        out = fopen ( "output.dat", "w" );
        p = Head;
        while ( p ) { // проход по списку и вывод результатов
                fprintf ( out, "%-20s\t%d\n", p->word, p->count );
                p = p->next;
                }
        fclose(out);
}
```

В переменной n хранится значение, которое вернула функция fscanf (количество удачно прочитанных элементов). Если это число меньше единицы (чтение прошло неудачно или закончились данные в файле), происходит выход из цикла while.

Сначала пытаемся искать это слово в списке с помощью функции Find. Если нашли –просто увеличиваем счетчик найденного узла. Если слово встретилось впервые, в памяти создается новый узел и заполняется данными. Затем с помощью функции FindPlace определяем, перед каким узлом списка надо его добавить.

Когда список готов, открываем файл для вывода и, используя стандартный проход по списку, выводим найденные слова и значения счетчиков.

Удаление узла

Эта процедура также связана с поиском заданного узла по всему списку, так как нам надо поменять ссылку у предыдущего узла, а перейти к нему непосредственно невозможно. Если мы нашли узел, за которым идет удаляемый узел, надо просто переставить ссылку.

Отдельно обрабатывается случай, когда удаляется первый элемент списка. При удалении узла освобождается память, которую он занимал.

Отдельно рассматриваем случай, когда удаляется первый элемент списка. В этом случае адрес удаляемого узла совпадает с адресом головы списка Head и надо просто записать в Head адрес следующего элемента.

```
void DeleteNode(PNode &Head, PNode OldNode)
{
        PNode q = Head;
        if (Head == OldNode)
        Head = OldNode->next; // удаляем первый элемент
        else {
                while (q && q->next != OldNode) // ищем элемент
                        q = q->next;
                if ( q == NULL ) return; // если не нашли, выход
                q->next = OldNode->next;
        }
        delete OldNode; // освобождаем память
}
```

Барьеры

Вы заметили, что для рассмотренного варианта списка требуется отдельно обрабатывать граничные случаи: добавление в начало, добавление в конец, удаление одного из крайних элементов. Можно значительно упростить приведенные выше процедуры, если установить два барьера – фиктивные первый и последний элементы. Таким образом, в списке всегда есть хотя бы два элемента-барьера, а все рабочие узлы находятся между ними.

Варианты заданий

Постановка задачи

В соответствии с вариантом разработать программу, которая содержит динамическую информацию в виде динамического односвязного списка.

Вариант 1

Составить программу, которая содержит динамическую информацию о наличии автобусов в автобусном парке. Сведения о каждом автобусе включают:

- номер автобуса;
- фамилию и инициалы водителя;
- номер маршрута.

Программа должна обеспечивать:

- начальное формирование данных обо всех автобусах в парке в виде односвязного списка;
- вывод всех автобусов;
- добавление автобуса в начало списка;
- добавление автобуса перед определенным автобусом;
- по запросу выдаются сведения об автобусах, находящихся в парке, или об автобусах, находящихся на маршруте.
- при выезде каждого автобуса из парка вводится номер автобуса, и программа удаляет данные об этом автобусе из списка автобусов, находящихся в парке, и записывает эти данные в список автобусов, находящихся на маршруте;
- при въезде каждого автобуса в парк вводится номер автобуса, и программа удаляет данные об этом автобусе из списка автобусов, находящихся на маршруте, и записывает эти данные в список автобусов, находящихся в парке;

Вариант 2

Составить программу, которая содержит текущую информацию о книгах в библиотеке. Сведения о книгах включают:

- номер УДК;
- фамилию и инициалы автора;
- название;
- год издания;
- количество экземпляров данной книги в библиотеке.

Программа должна обеспечивать:

- начальное формирование данных обо всех книгах в библиотеке в виде односвязного списка;
- добавление данных о книгах, вновь поступающих в библиотеку;
- удаление данных о списываемых книгах;
- добавление книги в начало списка;
- добавление книги в конец списка;
- добавление книги в позицию, отсортированную по имени в алфавитном порядке;
- по запросу выдаются сведения о наличии книг в библиотеке, упорядоченные по годам издания.

Вариант 3

Составить программу, которая содержит текущую информацию о заявках на авиабилеты. Каждая заявка включает:

- пункт назначения;
- номер рейса;
- фамилию и инициалы пассажира;
- желаемую дату вылета.

Программа должна обеспечивать:

- хранение всех заявок в виде односвязного списка;
- добавление заявок в список;
- удаление заявок;
- вывод заявок по заданному номеру рейса и дате вылета;
- вывод всех заявок.

Вариант 4

Составить программу, которая содержит текущую информацию о книгах в библиотеке. Сведения о книгах включают:

- номер УДК;
- фамилию и инициалы автора;
- название;
- год издания;
- количество экземпляров данной книги в библиотеке.

Программа должна обеспечивать:

- начальное формирование данных обо всех книгах в библиотеке в виде односвязного списка;
- добавление книги, отсортированной по автору;
- добавление книги перед указанной книгой;
- добавление книги после указанной книги.
- удаление выбранной книги.
- при выдаче каждой книги на руки вводится номер УДК, и программа уменьшает значение количества книг на единицу или выдает сообщение о том, что требуемой книги в библиотеке нет или требуемая книга находится на руках;
- при возвращении каждой книги вводится номер УДК, и программа увеличивает значение количества книг на единицу;
- по запросу выдаются сведения о наличии книг в библиотеке.

Вариант 5

Составить программу, которая содержит динамическую информацию о такси. Сведения о каждом такси включают:

- номер такси;
- марка автомобиля;
- фамилию и инициалы водителя;
- признак того, где находится такси — на вызове или в свободное.

Программа должна обеспечивать:

- начальное формирование данных обо всех такси в виде односвязного списка;
- вывод всех такси;

- добавление такси в начало списка;
- добавление такси перед определенным такси;
- удаление выбранного такси.
- при выезде каждого такси вводится номер такси, и программа устанавливает значение признака «такси на вызове»;
- при освобождении такси вводится номер такси, и программа устанавливает значение признака «такси свободное»;
- по запросу выдаются сведения о свободных такси, или о такси, находящихся на выезде.

Вариант 6

В файловой системе каталог файлов организован в виде односвязного списка. Для каждого файла в каталоге содержатся следующие сведения:

- имя файла;
- дата создания;
- количество обращений к файлу.

Написать программу, которая обеспечивает:

- начальное формирование каталога файлов;
- добавление файла перед указанным.
- добавление файла после указанного.
- вывод каталога файлов;
- удаление файлов, дата создания которых меньше заданной;
- выборку файла с наибольшим количеством обращений.

Программа должна обеспечивать диалог с помощью меню и контроль ошибок при вводе.

Вариант 7

Картотека в бюро обмена квартир организована в виде односвязного списка. Сведения о каждой квартире включают:

- количество комнат;
- этаж;
- площадь;
- адрес.

Написать программу, которая обеспечивает:

- начальное формирование картотеки;
- ввод заявки на обмен;
- поиск в картотеке подходящего варианта: при равенстве количества комнат и этажа и различии площадей в пределах 10% соответствующая карточка выводится и удаляется из списка, в противном случае поступившая заявка включается в список;
- вывод всего списка.
- удаление квартиры по адресу.
- добавление новой квартиры перед указанной в список.
- добавление новой квартиры после указанной.
- добавление квартиры отсортированной по адресу.

Программа должна обеспечивать диалог с помощью меню и контроль ошибок при вводе.

Вариант 8

Анкета для опроса населения содержит две группы вопросов. Первая группа содержит сведения о респонденте:

- возраст;
- пол;
- образование (начальное, среднее, высшее).

Вторая группа содержит собственно вопрос анкеты, ответом на который может являться либо ДА, либо НЕТ.

Написать программу, которая:

- обеспечивает начальный ввод анкет и формирует из них односвязного списка;
- на основе анализа анкет выдает ответы на следующие вопросы:

а) сколько мужчин старше 40 лет, имеющих высшее образование, ответили ДА на вопрос анкеты;

б) сколько женщин моложе 30 лет, имеющих среднее образование, ответили НЕТ на вопрос анкеты;

в) сколько мужчин моложе 25 лет, имеющих начальное образование, ответили ДА на вопрос анкеты;

- производит вывод всех анкет и ответов на вопросы.

Программа должна обеспечивать диалог с помощью меню и контроль ошибок при вводе.

Вариант 9

Предметный указатель организован в виде односвязного списка. Каждая компонента указателя содержит слово и номера страниц, на которых это слово встречается. Количество номеров страниц, относящихся к одному слову, лежит в диапазоне от одного до десяти. Написать программу, которая обеспечивает:

- начальное формирование предметного указателя;
- вывод предметного указателя;
- вывод номеров страниц для заданного слова.
- Поиск слова с максимальным количеством номером страниц.
- Добавление нового слова в начало списка.
- Добавление нового слова в конец списка.
- Добавление нового слова отсортированного по алфавиту.
- Добавление нового слова перед указанным словом.
- Добавление нового слова после указанного слова.
- Удаление указанного слова.

Программа должна обеспечивать диалог с помощью меню и контроль ошибок при вводе.

Вариант 10

Написать программу, которая содержит текущую информацию о книгах в библиотеке. Сведения о книгах включают:

- номер УДК;
- фамилию и инициалы автора;
- название;
- год издания;

- количество экземпляров данной книги в библиотеке.

Программа должна обеспечивать:

- начальное формирование данных о всех книгах в библиотеке в виде односвязного списка;
- добавление данных о книгах, вновь поступающих в библиотеку;
- удаление данных о списываемых книгах;
- Поиск книги с максимальным количеством экземпляров в библиотеке.
- Добавление новой книги в начало списка.
- Добавление новой книги в конец списка.
- Добавление новой книги отсортированной по названию в алфавитном порядке.
- Добавление новой книги перед указанной книгой.
- Добавление новой книги после указанной книгой.
- Удаление указанной книги.
- по запросу выдаются сведения о наличии книг в библиотеке, упорядоченные по годам издания.
-

Вариант 11

Текст помощи для некоторой программы организован в виде односвязного списка. Каждая компонента текста помощи содержит термин (слово) и текст, содержащий пояснения к этому термину. Количество строк текста, относящихся к одному термину, составляет от одной до пяти. Написать программу, которая обеспечивает:

- начальное формирование текста помощи;
- вывод текста помощи;
- вывод поясняющего текста для заданного термина.

Программа должна обеспечивать диалог с помощью меню и контроль ошибок при вводе.

Вариант 12

На междугородной телефонной станции картотека абонентов, содержащая сведения о телефонах и их владельцах, организована в виде односвязного списка. Написать программу, которая:

- обеспечивает начальное формирование картотеки в виде линейного списка;
- производит вывод всей картотеки;
- вводит номер телефона и время разговора;
- выводит извещение на оплату телефонного разговора.

Программа должна обеспечивать диалог с помощью меню и контроль ошибок при вводе.

Вариант 13

Автоматизированная информационная система на железнодорожном вокзале содержит сведения об отправлении поездов дальнего следования. Для каждого поезда указывается:

- номер поезда;
- станция назначения;
- время отправления.

Данные в информационной системе организованы в виде односвязного списка. Написать программу, которая:

- обеспечивает первоначальный ввод данных в информационную систему и формирование линейного списка;
- производит вывод всего списка;
- вводит номер поезда и выводит все данные об этом поезде;
- вводит название станции назначения и выводит данные обо всех поездах, следующих до этой станции.

Программа должна обеспечивать диалог с помощью меню и контроль ошибок при вводе.

Вариант 14

Гаражная стоянка имеет одну стояночную полосу, причем единственный въезд и единственный выезд находятся в одном конце полосы. Если владелец автомашины приходит забрать свой автомобиль, который не является ближайшим к выходу, то все автомашины, загораживающие проезд, удаляются, машина данного владельца выводится со стоянки, а другие машины возвращаются на стоянку в исходном порядке.

Написать программу, которая моделирует процесс прибытия и отъезда машин. Прибытие или отъезд автомашины задается командной строкой, которая содержит признак прибытия или отъезда и номер машины. Программа должна выводить сообщение при прибытии или выезде любой машины. При выезде автомашины со стоянки сообщение должно содержать число случаев, когда машина удалялась со стоянки для обеспечения выезда других автомобилей.

ЛАБОРАТОРНАЯ РАБОТА №15. ДИНАМИЧЕСКИЕ ДВУХСВЯЗНЫЕ СПИСКИ

Цель лабораторной работы

Разработка программ языке C++ с использованием динамических двухсвязных списков

Методические указания

Многие проблемы при работе с односвязным списком вызваны тем, что в них невозможно перейти к предыдущему элементу. Возникает естественная идея – хранить в памяти ссылку не только на следующий, но и на предыдущий элемент списка. Для доступа к списку используется не одна переменная-указатель, а две – ссылка на «голову» списка (**Head**) и на «хвост» - последний элемент (**Tail**).

Каждый узел содержит (кроме полезных данных) также ссылку на **следующий** за ним узел (поле **next**) и предыдущий (поле **prev**). Поле **next** у последнего элемента и поле **prev** у первого содержат **NULL**. Узел объявляется так:

```
struct Node {
        char word[40]; // область данных
        int count;
Node *next, *prev; // ссылки на соседние узлы
};
typedef Node *PNode; // тип данных «указатель на узел»
```

В дальнейшем мы будем считать, что указатель **Head** указывает на начало списка, а указатель **Tail** – на конец списка:

PNode Head = NULL, Tail = NULL;

Для пустого списка оба указателя равны **NULL**.

Операции с двусвязным списком

Добавление узла в начало списка

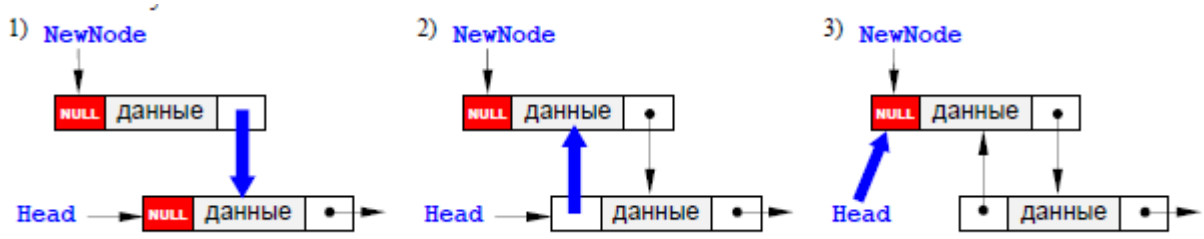
При добавлении нового узла **NewNode** в начало списка надо

1) установить ссылку **next** узла **NewNode** на голову существующего списка и его ссылку **prev** в **NULL**;

2) установить ссылку **prev** бывшего первого узла (если он существовал) на **NewNode**;

3) установить голову списка на новый узел;

4) если в списке не было ни одного элемента, хвост списка также устанавливается на новый узел.



По такой схеме работает следующая процедура:

```
void AddFirst(PNode &Head, PNode &Tail, PNode NewNode)
{
        NewNode->next = Head;
        NewNode->prev = NULL;
        if ( Head ) Head->prev = NewNode;
        Head = NewNode;
        if ( ! Tail ) Tail = Head; // этот элемент – первый
}
```

Добавление узла в конец списка

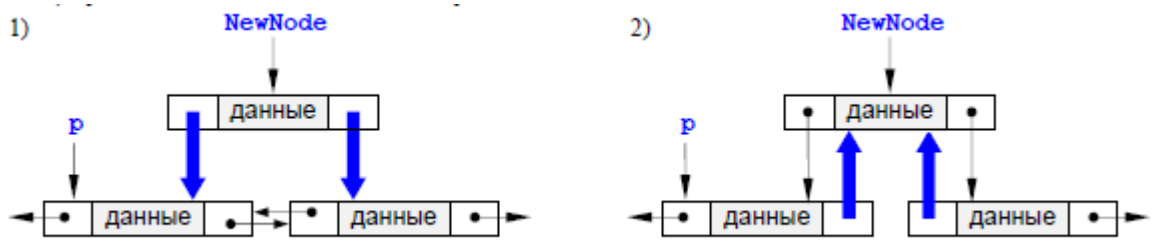
Благодаря симметрии добавление нового узла **NewNode** в конец списка проходит совершенно аналогично, в процедуре надо везде заменить **Head** на **Tail** и наоборот, а также поменять **prev** и **next**.

Добавление узла после заданного

Дан адрес **NewNode** нового узла и адрес **p** одного из существующих узлов в списке. Требуется вставить в список новый узел после **p**. Если узел **p** является последним, то операция сводится к добавлению в конец списка (см. выше). Если узел **p** – не последний, то операция вставки выполняется в два этапа:

1) установить ссылки нового узла на следующий за данным (***next***) и предшествующий ему (***prev***);

2) установить ссылки соседних узлов так, чтобы включить **NewNode** в список.



Такой метод реализует приведенная ниже процедура (она учитывает также возможность вставки элемента в конец списка, именно для этого в параметрах передаются ссылки на голову и хвост списка):

```
void AddAfter (PNode &Head, PNode &Tail, PNode p, PNode NewNode)
{
        if ( ! p->next )
                AddLast (Head, Tail, NewNode); // вставка в конец списка
else {
        NewNode->next = p->next; // меняем ссылки нового узла
        NewNode->prev = p;
```

```
            p->next->prev = NewNode; // меняем ссылки соседних узлов
            p->next = NewNode;
        }
    }
```

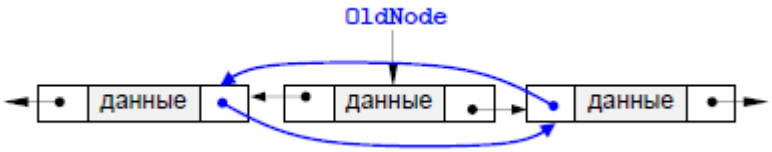
Добавление узла перед заданным выполняется аналогично.

Поиск узла в списке

Проход по двусвязному списку может выполняться в двух направлениях – от головы к хвосту (как для односвязного) или от хвоста к голове.

Удаление узла

Эта процедура также требует ссылки на голову и хвост списка, поскольку они могут измениться при удалении крайнего элемента списка. На первом этапе устанавливаются ссылки соседних узлов (если они есть) так, как если бы удаляемого узла не было бы. Затем узел удаляется и память, которую он занимает, освобождается. Эти этапы показаны на рисунке внизу. Отдельно проверяется, не является ли удаляемый узел первым или последним узлом списка.




```
void Delete(PNode &Head, PNode &Tail, PNode OldNode)
{
        if (Head == OldNode) {
                Head = OldNode->next; // удаляем первый элемент
                if ( Head )
                        Head->prev = NULL;
        else Tail = NULL; // удалили единственный элемент
}
else {
        OldNode->prev->next = OldNode->next;
        if ( OldNode->next )
        OldNode->next->prev = OldNode->prev;
        else Tail = NULL; // удалили последний элемент
        }
delete OldNode;
}
```

Варианты заданий

Постановка задачи

В соответствии с вариантом разработать программу, которая содержит динамическую информацию в виде динамического двухсвязного списка.

Вариант 1

Составить программу, которая содержит динамическую информацию о наличии автобусов в автобусном парке. Сведения о каждом автобусе включают:

- номер автобуса;
- фамилию и инициалы водителя;
- номер маршрута.

Программа должна обеспечивать:

- начальное формирование данных обо всех автобусах в парке в виде двухсвязного списка;
- вывод всех автобусов;
- добавление автобуса в начало списка;
- добавление автобуса перед определенным автобусом;
- по запросу выдаются сведения об автобусах, находящихся в парке, или об автобусах, находящихся на маршруте.
- при выезде каждого автобуса из парка вводится номер автобуса, и программа удаляет данные об этом автобусе из списка автобусов, находящихся в парке, и записывает эти данные в список автобусов, находящихся на маршруте;
- при въезде каждого автобуса в парк вводится номер автобуса, и программа удаляет данные об этом автобусе из списка автобусов, находящихся на маршруте, и записывает эти данные в список автобусов, находящихся в парке;

Вариант 2

Составить программу, которая содержит текущую информацию о книгах в библиотеке. Сведения о книгах включают:

- номер УДК;
- фамилию и инициалы автора;
- название;
- год издания;
- количество экземпляров данной книги в библиотеке.

Программа должна обеспечивать:

- начальное формирование данных обо всех книгах в библиотеке в виде двухсвязного списка;
- добавление данных о книгах, вновь поступающих в библиотеку;
- удаление данных о списываемых книгах;
- добавление книги в начало списка;
- добавление книги в конец списка;
- добавление книги в позицию, отсортированную по имени в алфавитном порядке;
- по запросу выдаются сведения о наличии книг в библиотеке, упорядоченные по годам издания.

Вариант 3

Составить программу, которая содержит текущую информацию о заявках на авиабилеты. Каждая заявка включает:

- пункт назначения;
- номер рейса;
- фамилию и инициалы пассажира;
- желаемую дату вылета.

Программа должна обеспечивать:

- хранение всех заявок в виде двухсвязного списка;

- добавление заявок в список;
- удаление заявок;
- вывод заявок по заданному номеру рейса и дате вылета;
- вывод всех заявок.

Вариант 4

Составить программу, которая содержит текущую информацию о книгах в библиотеке. Сведения о книгах включают:

- номер УДК;
- фамилию и инициалы автора;
- название;
- год издания;
- количество экземпляров данной книги в библиотеке.

Программа должна обеспечивать:

- начальное формирование данных обо всех книгах в библиотеке в виде двухсвязного списка;
- добавление книги, отсортированной по автору;
- добавление книги перед указанной книгой;
- добавление книги после указанной книги.
- удаление выбранной книги.
- при выдаче каждой книги на руки вводится номер УДК, и программа уменьшает значение количества книг на единицу или выдает сообщение о том, что требуемой книги в библиотеке нет или требуемая книга находится на руках;
- при возвращении каждой книги вводится номер УДК, и программа увеличивает значение количества книг на единицу;
- по запросу выдаются сведения о наличии книг в библиотеке.

Вариант 5

Составить программу, которая содержит динамическую информацию о такси. Сведения о каждом такси включают:

- номер такси;
- марка автомобиля;
- фамилию и инициалы водителя;
- признак того, где находится такси — на вызове или в свободное.

Программа должна обеспечивать:

- начальное формирование данных обо всех такси в виде двухсвязного списка;
- вывод всех такси;
- добавление такси в начало списка;
- добавление такси перед определенным такси;
- удаление выбранного такси.
- при выезде каждого такси вводится номер такси, и программа устанавливает значение признака «такси на вызове»;
- при освобождении такси вводится номер такси, и программа устанавливает значение признака «такси свободное»;
- по запросу выдаются сведения о свободных такси, или о такси, находящихся на выезде.

Вариант 6

В файловой системе каталог файлов организован в виде двухсвязного списка. Для каждого файла в каталоге содержатся следующие сведения:

- имя файла;
- дата создания;
- количество обращений к файлу.

Написать программу, которая обеспечивает:

- начальное формирование каталога файлов;
- добавление файла перед указанным.
- добавление файла после указанного.
- вывод каталога файлов;
- удаление файлов, дата создания которых меньше заданной;
- выборку файла с наибольшим количеством обращений.

Программа должна обеспечивать диалог с помощью меню и контроль ошибок при вводе.

Вариант 7

Картотека в бюро обмена квартир организована в виде двухсвязного списка. Сведения о каждой квартире включают:

- количество комнат;
- этаж;
- площадь;
- адрес.

Написать программу, которая обеспечивает:

- начальное формирование картотеки;
- ввод заявки на обмен;
- поиск в картотеке подходящего варианта: при равенстве количества комнат и этажа и различии площадей в пределах 10% соответствующая карточка выводится и удаляется из списка, в противном случае поступившая заявка включается в список;
- вывод всего списка.
- удаление квартиры по адресу.
- добавление новой квартиры перед указанной в список.
- добавление новой квартиры после указанной.
- добавление квартиры отсортированной по адресу.

Программа должна обеспечивать диалог с помощью меню и контроль ошибок при вводе.

Вариант 8

Анкета для опроса населения содержит две группы вопросов. Первая группа содержит сведения о респонденте:

- возраст;
- пол;
- образование (начальное, среднее, высшее).

Вторая группа содержит собственно вопрос анкеты, ответом на который может являться либо ДА, либо НЕТ.

Написать программу, которая:

- обеспечивает начальный ввод анкет и формирует из них двухсвязного списка;
- на основе анализа анкет выдает ответы на следующие вопросы:

а) сколько мужчин старше 40 лет, имеющих высшее образование, ответили ДА на вопрос анкеты;

б) сколько женщин моложе 30 лет, имеющих среднее образование, ответили НЕТ на вопрос анкеты;

в) сколько мужчин моложе 25 лет, имеющих начальное образование, ответили ДА на вопрос анкеты;

- производит вывод всех анкет и ответов на вопросы.

Программа должна обеспечивать диалог с помощью меню и контроль ошибок при вводе.

Вариант 9

Предметный указатель организован в виде двухсвязного списка. Каждая компонента указателя содержит слово и номера страниц, на которых это слово встречается. Количество номеров страниц, относящихся к одному слову, лежит в диапазоне от одного до десяти. Написать программу, которая обеспечивает:

- начальное формирование предметного указателя;
- вывод предметного указателя;
- вывод номеров страниц для заданного слова.
- Поиск слова с максимальным количеством номером страниц.
- Добавление нового слова в начало списка.
- Добавление нового слова в конец списка.
- Добавление нового слова отсортированного по алфавиту.
- Добавление нового слова перед указанным словом.
- Добавление нового слова после указанного слова.
- Удаление указанного слова.

Программа должна обеспечивать диалог с помощью меню и контроль ошибок при вводе.

Вариант 10

Написать программу, которая содержит текущую информацию о книгах в библиотеке. Сведения о книгах включают:

- номер УДК;
- фамилию и инициалы автора;
- название;
- год издания;
- количество экземпляров данной книги в библиотеке.

Программа должна обеспечивать:

- начальное формирование данных о всех книгах в библиотеке в виде двухсвязного списка;
- добавление данных о книгах, вновь поступающих в библиотеку;
- удаление данных о списываемых книгах;

- Поиск книги с максимальным количеством экземпляров в библиотеке.
- Добавление новой книги в начало списка.
- Добавление новой книги в конец списка.
- Добавление новой книги отсортированной по названию в алфавитном порядке.
- Добавление новой книги перед указанной книгой.
- Добавление новой книги после указанной книгой.
- Удаление указанной книги.
- по запросу выдаются сведения о наличии книг в библиотеке, упорядоченные по годам издания.
-

Вариант 11

Текст помощи для некоторой программы организован в виде двухсвязного списка. Каждая компонента текста помощи содержит термин (слово) и текст, содержащий пояснения к этому термину. Количество строк текста, относящихся к одному термину, составляет от одной до пяти. Написать программу, которая обеспечивает:

- начальное формирование текста помощи;
- вывод текста помощи;
- вывод поясняющего текста для заданного термина.

Программа должна обеспечивать диалог с помощью меню и контроль ошибок при вводе.

Вариант 12

На междугородной телефонной станции картотека абонентов, содержащая сведения о телефонах и их владельцах, организована в виде двухсвязного списка. Написать программу, которая:

- обеспечивает начальное формирование картотеки в виде линейного списка;
- производит вывод всей картотеки;
- вводит номер телефона и время разговора;
- выводит извещение на оплату телефонного разговора.

Программа должна обеспечивать диалог с помощью меню и контроль ошибок при вводе.

Вариант 13

Автоматизированная информационная система на железнодорожном вокзале содержит сведения об отправлении поездов дальнего следования. Для каждого поезда указывается:

- номер поезда;
- станция назначения;
- время отправления.

Данные в информационной системе организованы в виде двухсвязного списка. Написать программу, которая:

- обеспечивает первоначальный ввод данных в информационную систему и формирование линейного списка;

- производит вывод всего списка;
- вводит номер поезда и выводит все данные об этом поезде;
- вводит название станции назначения и выводит данные обо всех поездах, следующих до этой станции.

Программа должна обеспечивать диалог с помощью меню и контроль ошибок при вводе.

Вариант 14

Гаражная стоянка имеет одну стояночную полосу, причем единственный въезд и единственный выезд находятся в одном конце полосы. Если владелец автомашины приходит забрать свой автомобиль, который не является ближайшим к выходу, то все автомашины, загораживающие проезд, удаляются, машина данного владельца выводится со стоянки, а другие машины возвращаются на стоянку в исходном порядке.

Написать программу, которая моделирует процесс прибытия и отъезда машин. Прибытие или отъезд автомашины задается командной строкой, которая содержит признак прибытия или отъезда и номер машины. Программа должна выводить сообщение при прибытии или выезде любой машины. При выезде автомашины со стоянки сообщение должно содержать число случаев, когда машина удалялась со стоянки для обеспечения выезда других автомобилей.

ЛАБОРАТОРНАЯ РАБОТА №16. СТЭКИ И ОЧЕРЕДИ

Цель лабораторной работы

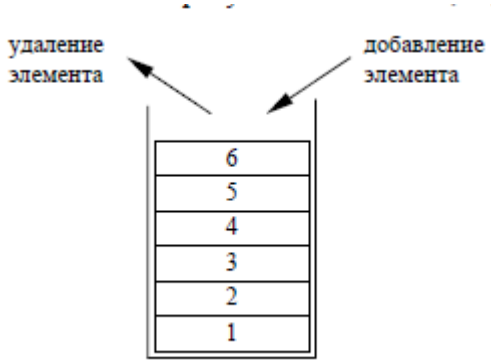
Получить практические навыки разработки программ на языке C++ с использованием динамических структрур данных в виде стеков и очередей.

Методические указания

Структуры, перечисленные в заголовке раздела, представляют собой списки, для которых разрешены только операции вставки и удаления первого и/или последнего элемента.

Стек

Стек – это упорядоченный набор элементов, в котором добавление новых и удаление существующих элементов допустимо только с одного конца, который называется **вершиной стека**.



Стек называют структурой типа ***LIFO*** (*Last In – First Out*) – последним пришел, первым ушел.

Стек похож на стопку с подносами, уложенными один на другой – чтобы достать какой-то поднос надо снять все подносы, которые лежат на нем, а положить новый поднос можно только сверху всей стопки. На рисунке показан стек, содержащий 6 элементов.

В современных компьютерах стек используется для

- размещения локальных переменных;
- размещения параметров процедуры или функции;
- сохранения адреса возврата (по какому адресу надо вернуться из процедуры);
- временного хранения данных, особенно при программировании на Ассемблере.

На стек выделяется ограниченная область памяти. При каждом вызове процедуры в стек добавляются новые элементы (параметры, локальные переменные, адрес возврата). Поэтому если вложенных вызовов будет много, стек переполнится. Очень опасной в отношении переполнения стека является **рекурсия**, поскольку она как раз и предполагает вложенные вызовы одной и той же процедуры или функции. При ошибке в программе рекурсия может стать бесконечной, кроме того, стек может переполниться, если вложенных вызовов будет слишком много.

Реализация стека с помощью массива

Если максимальный размер стека заранее известен, его можно реализовать в программе в виде массива. Удобно объединить в одной структуре сам массив и его размер. Объявим новый тип данных – стек на 100 элементов-символов.

```
const int MAXSIZE = 100;
```

```
struct Stack {
        char data[MAXSIZE];
        int size;
};
```

Для работы со стеком надо определить, как выполняются две операции – добавление элемента на вершину стека (**Push**) и снятие элемента с вершины стека (**Pop**).

```
void Push ( Stack &S, char x )
{
        if ( S.size == MAXSIZE ) {
                printf ("Стек переполнен");
                return;
        }
        S.data[S.size] = x;
        S.size ++;
}
```

Поскольку нумерация элементов массива **data** начинается с нуля, надо сначала записать новый элемент в **S.data[S.size]**, а затем увеличить размер стека. В процедуре предусмотрена обработка ошибки «переполнение стека». В этом случае на экран будет выдано сообщение «Стек переполнен». Можно также сделать функцию **Push**, которая будет возвращать 1 в случае удачного добавления элемента и 0 в случае ошибки.

Обратите внимание, что стек **S** передается в процедуру по ссылке, то есть, фактически передается адрес этого стека в памяти. Поэтому все операции со стеком в процедуре выполняются непосредственно со стеком вызывающей программы.

```
char Pop ( Stack &S )
{
        if ( S.size == 0 ) {
                printf ("Стек пуст");
                return char(255);
        }
        S.size --;
        return S.data[S.size];
}
```

Функция **Pop** возвращает символ, «снятый» с вершины стека, при этом размер стека уменьшается на единицу. Если стек пуст, функция возвращает символ с кодом 255 (который никогда не может находиться в стеке по условию задачи и сигнализирует об ошибке).

Задача. Ввести символьную строку, которая может содержать три вида скобок: (), [] и {}. Определить, верно ли расставлены скобки (символы между скобками не учитывать). Например, в строках **()[{}]** и **[{}([])]** скобки расставлены верно, а в строках **([)]** и **]]](((** - неверно.

Для одного вида скобок решение очень просто – ввести счетчик «вложенности» скобок, просмотреть всю строку, увеличивая счетчик для каждой открывающей скобки и уменьшая его для каждой закрывающей. Выражение записано верно, если счетчик ни разу не стал отрицательным и после обработки всей строки оказался равен нулю.

Если используются несколько видов скобок, счетчики не помогают. Однако эта задачи имеет красивое решение с помощью стека. Вначале стек пуст. Проходим всю

строку от начала до символа с кодом 0, который обозначает конец строки. Если встретили открывающую скобку, заносим ее в стек. Если встретили закрывающую скобку, то на вершине стека должна быть соответствующая ей открывающая скобка. Если это так, снимаем ее со стека. Если стек пуст или на вершине стека находится скобка другого вида, выражение неверное. В конце прохода стек должен быть пуст.

В приведенной ниже программе используются написанные ранее объявление структуры **Stack** и операции **Push** и **Pop**.

```
void main()
{
        char br1[3] = { '(', '[', '{' }; // открывающие скобки
        char br2[3] = { ')', ']', '}' }; // закрывающие скобки
        char s[80], upper;
        int i, k, OK;
        Stack S; // стек символов
        printf("Введите выражение со скобками> ");
        gets ( s );
        S.size = 0; // сначала стек пуст
        OK = 1;
        for (i = 0; OK && (s[i] != '\0'); i ++)
                for ( k = 0; k < 3; k ++ ) // проверить 3 вида скобок
                {
                        if ( s[i] == br1[k] ) { // открывающая скобка
                        Push ( S, s[i] ); break;
                }
                if ( s[i] == br2[k] ) { // закрывающая скобка
                        upper = Pop ( S );
                        if ( upper != br1[k] ) OK = 0;
                        break;
                }
        }
        if ( OK && (S.size == 0) )
        printf("\nВыражение правильное\n");
        else printf("\nВыражение неправильное \n");
}
```

Открывающие и закрывающие скобки записаны в массивах **br1** и **br2**. В самом начале стек пуст и его размер равен нулю (**S.size = 0**). Переменная **OK** служит для того, чтобы выйти из внешнего цикла, когда обнаружена ошибка (и не рассматривать оставшуюся часть строки). Она устанавливается в нуль, если в стеке обнаружена скобка другого типа или стек оказался пуст.

Реализация стека с помощью списка

Рассмотрим пример стека, в котором хранятся символы (это простейший вариант, элементом стека могут быть любые типы данных или структур, так же, как и для списка). Реализуем стек на основе двусвязного списка. При этом количество элементов стека ограничивается только доступным объемом памяти.

```
struct Node {
        char data;
```

```
        Node *next, *prev;
};
typedef Node *PNode;
```

Чтобы не работать с отдельными указателями на хвост и голову списка, объявим структуру, в которой будет храниться вся информация о стеке:

```
struct Stack {
        PNode Head, Tail;
};
```

В самом начале надо записать в обе ссылки стека **NULL**.

Добавление элемента на вершину стека

Фактически это добавление нового элемента в начало двусвязного списка. Эта процедура уже была написана ранее, теперь ее придется немного переделать, чтобы работать не с отдельными указателями, а со структурой типа **Stack**. В параметрах процедуры указывается не новый узел, а только *данные* для этого узла, то есть целое число. Память под новый узел выделяется в процедуре, то есть, скрыта от нас и снижает вероятность ошибки.

```
void Push ( Stack &S, char x )
{
        PNode NewNode;
        NewNode = new Node; // создать новый узел...
        NewNode->data = x; // и заполнить его данными
        NewNode->next = S.Head;
        NewNode->prev = NULL;
        if ( S.Head ) // добавить в начало списка
        S.Head->prev = NewNode;
        S.Head = NewNode;
        if ( ! S.Tail ) S.Tail = S.Head;
}
```

Получение верхнего элемента с вершины стека

Функция ***Pop*** удаляет верхний элемент и возвращает его данные.

```
char Pop ( Stack &S )
{
        PNode TopNode = S.Head;
        char x;
        if ( ! TopNode ) // если стек пуст, то
                return char(255); // вернуть символ с кодом 255
        x = TopNode->data;
        S.Head = TopNode->next;
        if ( S.Head ) S.Head->prev = NULL; // переставить ссылки
        else S.Tail = NULL;
        delete TopNode; // освободить память
        return x;
}
```

Системный стек в программах

При выполнении программ определенная область памяти отводится на стек программы.

Более того, в процессоре есть специальная ячейка (регистр), в которой хранится адрес вершины стека. Программа использует стек для хранения

1) адресов возврата из процедур и функций (это адреса, на которые переходит программа после выполнения процедуры или функции);

2) параметров, передаваемых в процедуры и функции;

3) локальных переменных в процедурах и функциях;

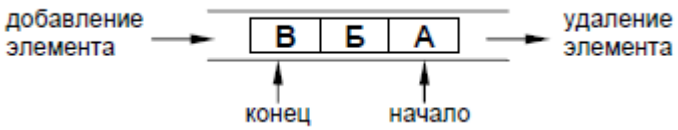
4) временных данных (в основном в программах на ассемблере).

Больше всего места занимают в стеке локальные переменные. Поэтому память под большие массивы надо выделять динамически. Кроме того, желательно не передавать в процедуры большие структуры, вместо этого можно передать их адрес или использовать передачу по ссылке (при этом перед именем параметра должен стоять знак **&**).

Очередь

Очередь – это упорядоченный набор элементов, в котором добавление новых элементов допустимо с одного конца (он называется **начало очереди**), а удаление существующих элементов – только с другого конца, который называется **концом очереди.**

Хорошо знакомой моделью является очередь в магазине. Очередь называют структурой типа ***FIFO*** (*First In – First Out*) – первым пришел, первым ушел. На рисунке изображена очередь из 3-х элементов.



Наиболее известные примеры применения очередей в программировании – очередь событий системы *Windows* и ей подобных. Очереди используются также для моделирования в **задачах массового обслуживания** (например, обслуживания клиентов в банке).

Реализация очереди с помощью массива

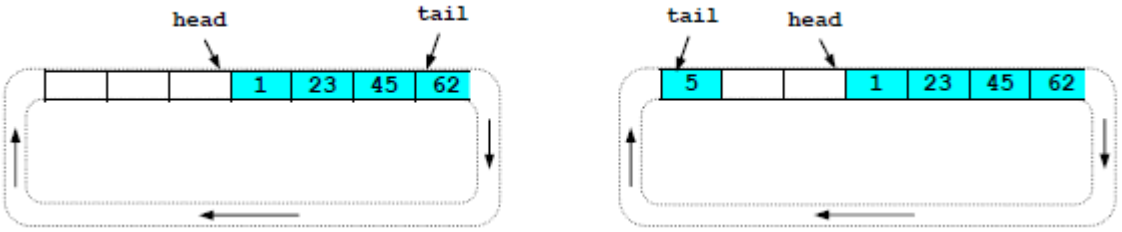
Если максимальный размер очереди заранее известен, его можно реализовать в программе в виде массива. Удобно объединить в одной структуре сам массив и его размер. Объявим новый тип данных – очередь на 100 элементов (целых чисел).

```
const int MAXSIZE = 100;
struct Queue {
        int data[MAXSIZE];
        int size, head, tail;
};
```

Если у стека один конец «закреплен» (не двигается), то у очереди «подвижны» оба конца. Что-бы не сдвигать все элементы в массиве при удалении или добавлении

элемента, обычно использую две переменные **head** и **tail** – первая из них обозначает номер первого элемента в очереди, а вторая – номер последнего. Если они равны, то в очереди всего один элемент. Массив как бы замыкается в кольцо – если массив закончился, но в начале массива есть свободные места, то новый элемент добавляется в начало массива, как показано на рисунках.

Для работы с очередью надо определить, как выполняются две операции – добавление элемента в конец очереди (**PushTail**) и удаление элемента с начала очереди (**Pop**).




```
void PushTail ( Queue &Q, int x )
{
        if ( Q.size == MAXSIZE ) {
                printf ("Очередь переполнена\n");
        return;
}
Q.tail++;
if ( Q.tail >= MAXSIZE ) // замыкание в кольцо
        Q.tail = 0;
Q.data[Q.tail] = x;
Q.size ++;
}
```

Поскольку очередь может начинаться не с начала массива (за счет того, что некоторые элементы уже «выбраны»), после увеличения **Q.tail** надо проверить, не вышли ли мы за границу массива. Если это случилось, новый элемент записывается в начало массива (хотя является хвостом очереди). В процедуре предусмотрена обработка ошибки «переполнение очереди». В этом случае на экран будет выдано сообщение «Очередь переполнена». Можно также сделать функцию **PushTail**, которая будет возвращать 1 в случае удачного добавления элемента и 0 в случае ошибки.

Обратите внимание, что очередь ***Q*** передается в процедуру по ссылке, то есть, фактически передается адрес этой структуры в памяти.

```
int Pop ( Queue &Q )
{
        int temp;
        if ( Q.size == 0 ) {
                printf ("Очередь пуста\n");
                return 32767; // сигнал об ошибке
        }
        temp = Q.data[Q.head];
        Q.head ++;
        if ( Q.head >= MAXSIZE ) Q.head = 0;
        Q.size --;
```

```
        return temp;
}
```

Функция **Pop** возвращает число, полученное с начала очереди, при этом размер очереди уменьшается на единицу. Если стек пуст, функция возвращает число 32767 (предполагается, что оно не может находиться в очереди по условию задачи и сигнализирует об ошибке).

Реализация очереди с помощью списка

Если максимальный размер заранее неизвестен или требуется сделать его динамическим, для реализации используют список. Рассмотрим пример очереди, элементами которой являются целые числа. При этом количество ее элементов ограничивается только доступным объемом памяти. Новые типы данных (узел и указатель на него) объявляются так же, как для списка:

```
struct Node {
        int data;
        Node *next, *prev;
};
typedef Node *PNode;
```

Чтобы не работать с отдельными указателями на хвост и голову списка, объявим структуру, в которой будет храниться вся информация об очереди:

```
struct Queue {
        PNode head, tail;
};
```

В самом начале надо записать в обе ссылки **NULL**. Заметим, что такая же структура использовалась и для стека. Более того, функция для получения первого элемента очереди (**Pop**) совпадает с функцией снятия элемента с вершины стека (напишите ее в качестве упражнения).

Добавление элемента в конец очереди

Фактически это добавление нового элемента в конец двусвязного списка. В параметрах процедуры указывается не новый узел, а только *данные* для этого узла, то есть целое число.

Память под новый узел выделяется в процедуре, то есть, скрыта от нас и снижает вероятность ошибки.

```
void PushTail ( Queue &Q, int x )
{
        PNode NewNode;
        NewNode = new Node; // создать новый узел
        NewNode->data = x; // заполнить узел данными
        NewNode->prev = Q.Tail;
        NewNode->next = NULL;
        if ( Q.tail ) // добавить узел в конец списка
                Q.tail->next = NewNode;
```

```
    Q.tail = NewNode;
    if ( ! Q.head ) Q.head = Q.tail;
}
```

Дек

Дек (*deque*) - это упорядоченный набор элементов, в котором добавление новых и удаление существующих элементов допустимо с любого конца.

Дек может быть реализован на основе массива или двусвязного списка. Для дека разрешены четыре операции:

1) добавление элемента в начало;
2) добавление элемента в конец;
3) удаление элемента с начала;
4) удаление элемента с конца.

Их можно реализовать, используя написанные выше процедуры для стека и очереди.

Варианты заданий

Постановка задания.

Задача 1. Решить задачу в виде стека, убрав действия, которые невозможно выполнить над данной структурой.

Задача 2. Решить задачу в виде очереди, убрав действия, которые невозможно выполнить над данной структурой.

Вариант 1

Составить программу, которая содержит динамическую информацию о наличии автобусов в автобусном парке. Сведения о каждом автобусе включают:

- номер автобуса;
- фамилию и инициалы водителя;
- номер маршрута.

Программа должна обеспечивать:

- начальное формирование данных обо всех автобусах в парке в виде стека или очереди;
- вывод всех автобусов;
- добавление автобуса в начало списка;
- добавление автобуса перед определенным автобусом;
- по запросу выдаются сведения об автобусах, находящихся в парке, или об автобусах, находящихся на маршруте.
- при выезде каждого автобуса из парка вводится номер автобуса, и программа удаляет данные об этом автобусе из списка автобусов, находящихся в парке, и записывает эти данные в список автобусов, находящихся на маршруте;
- при въезде каждого автобуса в парк вводится номер автобуса, и программа удаляет данные об этом автобусе из списка автобусов, находящихся на маршруте, и записывает эти данные в список автобусов, находящихся в парке;

Вариант 2

Составить программу, которая содержит текущую информацию о книгах в библиотеке. Сведения о книгах включают:

- номер УДК;

- фамилию и инициалы автора;
- название;
- год издания;
- количество экземпляров данной книги в библиотеке.

Программа должна обеспечивать:

- начальное формирование данных обо всех книгах в библиотеке в виде стека или очереди;
- добавление данных о книгах, вновь поступающих в библиотеку;
- удаление данных о списываемых книгах;
- добавление книги в начало списка;
- добавление книги в конец списка;
- добавление книги в позицию, отсортированную по имени в алфавитном порядке;
- по запросу выдаются сведения о наличии книг в библиотеке, упорядоченные по годам издания.

Вариант 3

Составить программу, которая содержит текущую информацию о заявках на авиабилеты. Каждая заявка включает:

- пункт назначения;
- номер рейса;
- фамилию и инициалы пассажира;
- желаемую дату вылета.

Программа должна обеспечивать:

- хранение всех заявок в виде стека или очереди;
- добавление заявок в список;
- удаление заявок;
- вывод заявок по заданному номеру рейса и дате вылета;
- вывод всех заявок.

Вариант 4

Составить программу, которая содержит текущую информацию о книгах в библиотеке. Сведения о книгах включают:

- номер УДК;
- фамилию и инициалы автора;
- название;
- год издания;
- количество экземпляров данной книги в библиотеке.

Программа должна обеспечивать:

- начальное формирование данных обо всех книгах в библиотеке в виде стека или очереди;
- добавление книги, отсортированной по автору;
- добавление книги перед указанной книгой;
- добавление книги после указанной книги.
- удаление выбранной книги.
- при выдаче каждой книги на руки вводится номер УДК, и программа уменьшает значение количества книг на единицу или выдает сообщение о том, что требуемой книги в библиотеке нет или требуемая книга находится на руках;

- при возвращении каждой книги вводится номер УДК, и программа увеличивает значение количества книг на единицу;
- по запросу выдаются сведения о наличии книг в библиотеке.

Вариант 5

Составить программу, которая содержит динамическую информацию о такси. Сведения о каждом такси включают:

- номер такси;
- марка автомобиля;
- фамилию и инициалы водителя;
- признак того, где находится такси — на вызове или в свободное.

Программа должна обеспечивать:

- начальное формирование данных обо всех такси в виде стека или очереди;
- вывод всех такси;
- добавление такси в начало списка;
- добавление такси перед определенным такси;
- удаление выбранного такси.
- при выезде каждого такси вводится номер такси, и программа устанавливает значение признака «такси на вызове»;
- при освобождении такси вводится номер такси, и программа устанавливает значение признака «такси свободное»;
- по запросу выдаются сведения о свободных такси, или о такси, находящихся на выезде.

Вариант 6

В файловой системе каталог файлов организован в виде стека или очереди. Для каждого файла в каталоге содержатся следующие сведения:

- имя файла;
- дата создания;
- количество обращений к файлу.

Написать программу, которая обеспечивает:

- начальное формирование каталога файлов;
- добавление файла перед указанным.
- добавление файла после указанного.
- вывод каталога файлов;
- удаление файлов, дата создания которых меньше заданной;
- выборку файла с наибольшим количеством обращений.

Программа должна обеспечивать диалог с помощью меню и контроль ошибок при вводе.

Вариант 7

Картотека в бюро обмена квартир организована в виде стека или очереди. Сведения о каждой квартире включают:

- количество комнат;
- этаж;
- площадь;

- адрес.

Написать программу, которая обеспечивает:

- начальное формирование картотеки;
- ввод заявки на обмен;
- поиск в картотеке подходящего варианта: при равенстве количества комнат и этажа и различии площадей в пределах 10% соответствующая карточка выводится и удаляется из списка, в противном случае поступившая заявка включается в список;
- вывод всего списка.
- удаление квартиры по адресу.
- добавление новой квартиры перед указанной в список.
- добавление новой квартиры после указанной.
- добавление квартиры отсортированной по адресу.

Программа должна обеспечивать диалог с помощью меню и контроль ошибок при вводе.

Вариант 8

Анкета для опроса населения содержит две группы вопросов. Первая группа содержит сведения о респонденте:

- возраст;
- пол;
- образование (начальное, среднее, высшее).

Вторая группа содержит собственно вопрос анкеты, ответом на который может являться либо ДА, либо НЕТ.

Написать программу, которая:

- обеспечивает начальный ввод анкет и формирует из них стека или очереди;
- на основе анализа анкет выдает ответы на следующие вопросы:

а) сколько мужчин старше 40 лет, имеющих высшее образование, ответили ДА на вопрос анкеты;

б) сколько женщин моложе 30 лет, имеющих среднее образование, ответили НЕТ на вопрос анкеты;

в) сколько мужчин моложе 25 лет, имеющих начальное образование, ответили ДА на вопрос анкеты;

- производит вывод всех анкет и ответов на вопросы.

Программа должна обеспечивать диалог с помощью меню и контроль ошибок при вводе.

Вариант 9

Предметный указатель организован в виде стека или очереди. Каждая компонента указателя содержит слово и номера страниц, на которых это слово встречается. Количество номеров страниц, относящихся к одному слову, лежит в диапазоне от одного до десяти. Написать программу, которая обеспечивает:

- начальное формирование предметного указателя;
- вывод предметного указателя;
- вывод номеров страниц для заданного слова.
- Поиск слова с максимальным количеством номером страниц.

- Добавление нового слова в начало списка.
- Добавление нового слова в конец списка.
- Добавление нового слова отсортированного по алфавиту.
- Добавление нового слова перед указанным словом.
- Добавление нового слова после указанного слова.
- Удаление указанного слова.

Программа должна обеспечивать диалог с помощью меню и контроль ошибок при вводе.

Вариант 10

Написать программу, которая содержит текущую информацию о книгах в библиотеке. Сведения о книгах включают:

- номер УДК;
- фамилию и инициалы автора;
- название;
- год издания;
- количество экземпляров данной книги в библиотеке.

Программа должна обеспечивать:

- начальное формирование данных о всех книгах в библиотеке в виде стека или очереди;
- добавление данных о книгах, вновь поступающих в библиотеку;
- удаление данных о списываемых книгах;
- Поиск книги с максимальным количеством экземпляров в библиотеке.
- Добавление новой книги в начало списка.
- Добавление новой книги в конец списка.
- Добавление новой книги отсортированной по названию в алфавитном порядке.
- Добавление новой книги перед указанной книгой.
- Добавление новой книги после указанной книгой.
- Удаление указанной книги.
- по запросу выдаются сведения о наличии книг в библиотеке, упорядоченные по годам издания.
-

Вариант 11

Текст помощи для некоторой программы организован в виде стека или очереди. Каждая компонента текста помощи содержит термин (слово) и текст, содержащий пояснения к этому термину. Количество строк текста, относящихся к одному термину, составляет от одной до пяти. Написать программу, которая обеспечивает:

- начальное формирование текста помощи;
- вывод текста помощи;
- вывод поясняющего текста для заданного термина.

Программа должна обеспечивать диалог с помощью меню и контроль ошибок при вводе.

Вариант 12

На междугородной телефонной станции картотека абонентов, содержащая сведения о телефонах и их владельцах, организована в виде стека или очереди. Написать программу, которая:

- обеспечивает начальное формирование картотеки в виде линейного списка;
- производит вывод всей картотеки;
- вводит номер телефона и время разговора;
- выводит извещение на оплату телефонного разговора.

Программа должна обеспечивать диалог с помощью меню и контроль ошибок при вводе.

Вариант 13

Автоматизированная информационная система на железнодорожном вокзале содержит сведения об отправлении поездов дальнего следования. Для каждого поезда указывается:

- номер поезда;
- станция назначения;
- время отправления.

Данные в информационной системе организованы в виде стека или очереди. Написать программу, которая:

- обеспечивает первоначальный ввод данных в информационную систему и формирование линейного списка;
- производит вывод всего списка;
- вводит номер поезда и выводит все данные об этом поезде;
- вводит название станции назначения и выводит данные обо всех поездах, следующих до этой станции.

Программа должна обеспечивать диалог с помощью меню и контроль ошибок при вводе.

Вариант 14

Гаражная стоянка имеет одну стояночную полосу, причем единственный въезд и единственный выезд находятся в одном конце полосы. Если владелец автомашины приходит забрать свой автомобиль, который не является ближайшим к выходу, то все автомашины, загораживающие проезд, удаляются, машина данного владельца выводится со стоянки, а другие машины возвращаются на стоянку в исходном порядке.

Написать программу, которая моделирует процесс прибытия и отъезда машин. Прибытие или отъезд автомашины задается командной строкой, которая содержит признак прибытия или отъезда и номер машины. Программа должна выводить сообщение при прибытии или выезде любой машины. При выезде автомашины со стоянки сообщение должно содержать число случаев, когда машина удалялась со стоянки для обеспечения выезда других автомобилей.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Дейтел П.Дж. Как программировать на C: 4-е издание / Дейтел П.Дж., Дейтел Х.М. – М.: Издательство «Бином-Пресс», 2009. – 1002 с.
2. Лафоре, Р. Объектно-ориентированное программирование в C++ , 4- е изд.: Пер. с англ. — М. : Издательский дом "Питер", 2004. – 922 с.
3. Красновидов, А.В. Теория языков программирования и методы трансляции [Текст] : учеб. пособие / А. В. Красновидов ; Учеб.-метод. центр по образованию на ж.-д. трансп. - М. : [б. и.], 2016. – 176 с.
4. Павловская, Т.А. C/C++. Программирование на языке высокого уровня : учеб. для вузов/ Т.А. Павловская. -М.; СПб.: Питер, 2006. – 460 с.
5. Программирование на языке высокого уровня C/C++ [Текст] / сост. С. П. Зоткин. – 2016. – 140 с. http://www.iprbookshop.ru/48037.html.
6. Страуструп, Б. «Язык программирования C++». – М.;СПб. : «Издательство БИНОМ» – «Невский диалект», 2001. – 1099 с.
7. Страуструп, Б. «Программирование: принципы и практика использования C++». – М.;ООО «И.Д. Вильямс», 2011. – 1248 с.
8. Шилдт, Г. C++: руководство для начинающих, 2-е издание. : Пер. с англ. – М. : Издательский дом "Вильямс", 2005. – 672 с.

Учебное издание

Игнатьева Олеся Владимировна

ИНФОРМАТИКА И ПРОГРАММИРОВАНИЕ

Часть 2

Печатается в авторской редакции
Технический редактор Н.С. Федорова

Подписано в печать 03.10.17. Формат 60×84/16.
Бумага газетная. Ризография. Усл. печ. л. 9,5.
Тираж экз. Изд. № 9078. Заказ .

Редакционно-издательский центр ФГБОУ ВО РГУПС.

Адрес университета: 344038, г. Ростов н/Д, пл. Ростовского Стрелкового Полка Народного Ополчения, д. 2.