Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное учреждение
высшего профессионального образования

«ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
СИСТЕМ УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ»
(ТУСУР)

Кафедра моделирования и системного анализа

Ганджа Т.В., Панов С.А.

# ОБЪЕКТНО-ОРИЕНТИРОВАННОЕ ПРОГРАММИРОВАНИЕ

Методические указания к лабораторным работам

Томск
2015

Ганджа Т.В., Панов С.А. Объектно-ориентированное программирование / Методические указания к лабораторным работам – Томск: Томский государственный университет систем управления и радиоэлектроники, Кафедра моделирования и системного анализа, 2015. – 102 с.

**Содержание**

## Введение

Курс «Объектно-ориентированное программирование» рассматривает как базовые конструкции языка программирования C++, так и расширенные функции по работе с файлами, динамическими структурами данных и классами. Задача курса состоит в изучении структурного, модульного и объектно-ориентированного программирования.

Задачей настоящих методических указаний является помощь студентам очной формы обучения при изучении курса «Объектно-ориентированное программирование».

## Перечень лабораторных работ

**Лабораторная работа № 1. (4 часа).** Циклические вычислительные процедуры.

**Лабораторная работа № 2 (4 часа).** Указатели и одномерные массивы.

**Лабораторная работа № 3 (4 часа).** Многомерные массивы.

**Лабораторная работа № 4 (4 часа).** Структуры.

**Лабораторная работа № 5 (4 часа).** Простейшие функции.

**Лабораторная работа № 6 (4 часов).** Функции работы со строками.

**Лабораторная работа № 7 (4 часа).** Создание и использование классов.

**Лабораторная работа № 8 (4 часа).** Наследование.

## Лабораторная работа № 1. Циклические вычислительные процедуры

Цель работы – отработка умения и навыков работы с базовыми операциями компиляции и компоновки программ, написанных на языке программирования C++; получение навыков использования условных операторов и циклических вычислительных процедур.

### Методические указания

Для выполнения лабораторной работы необходимо изучить следующие разделы курса лекций:

- 1.1 Структура программы;
- 1.2 Операции;
- 1.3 Выражения;
- 1.4 Базовые конструкции структурного программирования.

Особое внимание при выполнении лабораторной работы следует уделить условному оператору *if* и циклу с параметром *for*. Условный оператор *if* используется для разветвления процесса вычислений на два направления. Формат оператора следующий:

*if (выражение) оператор_1; [else оператор_2;]*

Если *выражение* не равно нулю (или имеет значение *true*) выполняется *оператор_1*, иначе – *оператор_2*.

Цикл с параметром имеет следующий формат:

*for (инициализация: выражение; модификации) оператор;*

*Инициализация* используется для объявления и присвоения начальных значений величинам, используемым в цикле. В этой части можно записать несколько операторов, разделенных запятой. *Модификации* выполняются после каждой итерации цикла и служат обычно для изменения параметров цикла. В части модификаций можно записать несколько операторов через запятую. Простой или составной *оператор* представляет собой тело цикла. Любая из частей оператора *for* может быть опущена (кроме точки с запятой).

## Пример выполнения лабораторной работы

*Задание:*

Вычислить и вывести на экран в виде таблицы значения функции $F$ на интервале от $X_{нач}$ до $X_{кон}$ с шагом $dX$.

$$F(x)=\begin{cases} a\cdot x^2+b & \text{при } x<0 \text{ и } b\neq 0 \\ \dfrac{x-a}{x-c} & \text{при } x>0 \text{ и } b=0 \\ \dfrac{x}{c} & \text{в остальных случаях} \end{cases}$$

где $a$, $b$, $c$ — действительные числа.

Значения $a$, $b$, c, $X_{нач}$, $X_{кон}$, $dX$ ввести с клавиатуры. Предусмотреть недопущения нестандартных ситуаций.

*Листинг программы:*

```
void create_table(int a, int b, int c, int Xstart, int Xend, int dX, int Fx);
void insert_line();

void create_table(int a, int b, int c, int Xstart, int Xend, int dX, int Fx)
{
	printf("| %d | %d | %d |    %d    |  %d  |  %d |   %d    |\n", a, b, c, Xstart,
Xend, dX, Fx);
	insert_line();

}

void insert_line()
{
	printf("----------------------------------------\n");
}

int main ()
{
	int a,b,c,Xstart,Xend,dX, Fx;
	printf("Enter: a, b, c, X start, X end, dX: \n");
	scanf_s ("\n %d, %d, %d, %d, %d, %d",&a,&b,&c,&Xstart,&Xend,&dX);
	insert_line();
	printf("| a | b | c | Xstart | Xend | dX | F(x) |\n");
	insert_line();
	for(int x=Xstart;x<Xend;x=x+dX)
	{
		if(x<0 && b!=0)
			Fx = (a*x*x)+b;
		else if(x>0 && b==0)
		{
			if((x-c)>0)
				Fx = (x-a)/(x-c);
			else
			{
```

```
                        printf("Error! Divizion by zero!");
                        _gettch();
                    }
                }
                else
                    Fx = x/c;
                if (Fx!=0)
                    create_table(a, b, c, Xstart, Xend, dX, Fx);
            }
            _gettch();
            return 0;
}
```

## Варианты заданий

### Вариант 1

Вычислить и вывести на экран в виде таблицы значения функции $F$ на интервале от $X_{нач}$ до $X_{кон}$ с шагом $dX$.

$$F(x)=\begin{cases} a\cdot x^2+b & \text{при } x<0 \text{ и } b\neq 0 \\ \dfrac{x-a}{x-c} & при \quad x>0 \ и \ b=0 \\ \dfrac{x}{c} & в\ остальных\ случаях \end{cases}$$

где $a$, $b$, $c$ — действительные числа.

Значения $a$, $b$, c, $X_{нач}$, $X_{кон}$, $dX$ ввести с клавиатуры. Предусмотреть недопущения нестандартных ситуаций.

### Вариант 2

Вычислить и вывести на экран в виде таблицы значения функции $F$ на интервале от $X_{нач}$ до $X_{кон}$ с шагом $dX$.

$$F(x)=\begin{cases} \dfrac{1}{ax}-b & при \quad x+5<0 \ и \ c=0 \\ \dfrac{x-a}{x} & при \quad x+5>0 \ и \ c\neq 0 \\ \dfrac{10\cdot x}{c-4} & в\ остальных\ случаях \end{cases}$$

где $a$, $b$, $c$ — действительные числа.

Значения *a, b,* c, $X_{нач}$, $X_{кон}$, *dX* ввести с клавиатуры. Предусмотреть недопущения нестандартных ситуаций.

Вариант 3

Вычислить и вывести на экран в виде таблицы значения функции *F* на интервале от $X_{нач}$ до $X_{кон}$ с шагом *dX*.

$$F(x)=\begin{cases} a\cdot x^2+b\cdot x+c & при\ a<0\ и\ c\neq 0 \\ \dfrac{-a}{x-c} & при\ \ a>0\ \ и\ c=0 \\ a\cdot(x+c) & в\ остальных\ случаях \end{cases}$$

где *a, b, c* — действительные числа.

Значения *a, b,* c, $X_{нач}$, $X_{кон}$, *dX* ввести с клавиатуры. Предусмотреть недопущения нестандартных ситуаций.

Вариант 4

Вычислить и вывести на экран в виде таблицы значения функции *F* на интервале от $X_{нач}$ до $X_{кон}$ с шагом *dX*.

$$F(x)=\begin{cases} -a\cdot x-c & при\ c<0\ и\ x\neq 0 \\ \dfrac{x-a}{-c} & при\ \ c>0\ \ и\ x=0 \\ \dfrac{bx}{c-a} & в\ остальных\ случаях \end{cases}$$

где *a, b, c* — действительные числа.

Значения *a, b,* c, $X_{нач}$, $X_{кон}$, *dX* ввести с клавиатуры. Предусмотреть недопущения нестандартных ситуаций.

Вариант 5

Вычислить и вывести на экран в виде таблицы значения функции *F* на интервале от $X_{нач}$ до $X_{кон}$ с шагом *dX*.

$$F(x)=\begin{cases} a-\dfrac{x}{10+b} & при\ x<0\ и\ b\neq 0 \\ \dfrac{x-a}{x-c} & при\ \ x>0\ \ и\ b=0 \\ 3\cdot x+\dfrac{2}{c} & в\ остальных\ случаях \end{cases}$$

где *a, b, c* — действительные числа.

Значения *a, b,* с, $X_{нач}$, $X_{кон}$, *dX* ввести с клавиатуры. Предусмотреть недопущения нестандартных ситуаций.

Вариант 6

Вычислить и вывести на экран в виде таблицы значения функции *F* на интервале от $X_{нач}$ до $X_{кон}$ с шагом *dX*.

$$F(x)=\begin{cases} a\cdot x^2+b^2\cdot x & при\ c<0\ и\ b\neq 0 \\ \dfrac{x+a}{x+c} & при\ \ c>0\ \ и\ b=0 \\ \dfrac{x}{c} & в\ остальных\ случаях \end{cases}$$

где *a, b, c* — действительные числа.

Значения *a, b,* с, $X_{нач}$, $X_{кон}$, *dX* ввести с клавиатуры. Предусмотреть недопущения нестандартных ситуаций.

Вариант 7

Вычислить и вывести на экран в виде таблицы значения функции *F* на интервале от $X_{нач}$ до $X_{кон}$ с шагом *dX*.

$$F(x)=\begin{cases} -a\cdot x^2-b & при\ x<5\ и\ c\neq 0 \\ \dfrac{x-a}{x} & при\ \ x>5\ \ и\ c=0 \\ \dfrac{-x}{c} & в\ остальных\ случаях \end{cases}$$

где *a, b, c* — действительные числа.

Значения *a, b,* с, $X_{нач}$, $X_{кон}$, *dX* ввести с клавиатуры. Предусмотреть недопущения нестандартных ситуаций.

Вариант 8

Вычислить и вывести на экран в виде таблицы значения функции *F* на интервале от $X_{нач}$ до $X_{кон}$ с шагом *dX*.

$$F(x)=\begin{cases} -a\cdot x^2 & при\ \ c<5\ и\ \text{a}\neq 0 \\ \dfrac{a-x}{c\cdot x} & при\ \ \ x>5\ \ и\ \ c=0 \\ \dfrac{x}{c} & в\ остальных\ случаях \end{cases}$$

где *a, b, c* — действительные числа.

Значения *a, b,* с, $X_{нач}$, $X_{кон}$, *dX* ввести с клавиатуры. Предусмотреть недопущения нестандартных ситуаций.

Вариант 9

Вычислить и вывести на экран в виде таблицы значения функции *F* на интервале от $X_{нач}$ до $X_{кон}$ с шагом *dX*.

$$F(x)=\begin{cases} a\cdot x^2+b^2\cdot x & при\ \ a<0\ и\ x\neq 0 \\ x-\dfrac{a}{x-c} & при\ \ \ a>0\ \ и\ \ x=0 \\ 1+\dfrac{x}{c} & в\ остальных\ случаях \end{cases}$$

где *a, b, c* — действительные числа.

Значения *a, b,* с, $X_{нач}$, $X_{кон}$, *dX* ввести с клавиатуры. Предусмотреть недопущения нестандартных ситуаций.

Вариант 10

Вычислить и вывести на экран в виде таблицы значения функции *F* на интервале от $X_{нач}$ до $X_{кон}$ с шагом *dX*.

$$F(x)=\begin{cases} a\cdot x^2-b\cdot x+c \quad при\ x<3\ и\ b\neq 0 \\ \dfrac{x-a}{x-c} \quad при\ \ x>3\ \ и\ b=0 \\ \dfrac{x}{c}\ \ в\ остальных\ случаях \end{cases}$$

где *a, b, c* — действительные числа.

Значения *a, b,* с, $X_{нач}$, $X_{кон}$, *dX* ввести с клавиатуры. Предусмотреть недопущения нестандартных ситуаций.

Вариант 11

Вычислить и вывести на экран в виде таблицы значения функции *F* на интервале от $X_{нач}$ до $X_{кон}$ с шагом *dX*.

$$F(x)=\begin{cases} a\cdot x^2+\dfrac{b}{c} \quad при\ x<1\ и\ c\neq 0 \\ \dfrac{x-a}{(x-c)^2} \quad при\ \ x>1.5\ \ и\ c=0 \\ \dfrac{x^2}{c^2}\ \ в\ остальных\ случаях \end{cases}$$

где *a, b, c* — действительные числа.

Значения *a, b,* с, $X_{нач}$, $X_{кон}$, *dX* ввести с клавиатуры. Предусмотреть недопущения нестандартных ситуаций.

Вариант 12

Вычислить и вывести на экран в виде таблицы значения функции *F* на интервале от $X_{нач}$ до $X_{кон}$ с шагом *dX*.

$$F(x)=\begin{cases} a\cdot x^3+b^2+c \quad при\ x<0.6\ и\ b+c\neq 0 \\ \dfrac{x-a}{x-c} \quad при\ \ x>0.6\ \ и\ b+c=0 \\ \dfrac{x}{c}+\dfrac{x}{a}\ \ в\ остальных\ случаях \end{cases}$$

где *a, b, c* — действительные числа.

Значения *a, b,* с, $X_{нач}$, $X_{кон}$, *dX* ввести с клавиатуры. Предусмотреть недопущения нестандартных ситуаций.

Вариант 13

Вычислить и вывести на экран в виде таблицы значения функции *F* на интервале от $X_{нач}$ до $X_{кон}$ с шагом *dX*.

$$F(x)=\begin{cases} a\cdot x^2+b \quad при\ x-1<0\ и\ b-x\neq 0 \\ \dfrac{x-a}{x} \quad при\quad x-1>0\ и\ b+x=0 \\ \dfrac{x}{c}\ в\ остальных\ случаях \end{cases}$$

где *a, b, c* — действительные числа.

Значения *a, b,* с, $X_{нач}$, $X_{кон}$, *dX* ввести с клавиатуры. Предусмотреть недопущения нестандартных ситуаций.

Вариант 14

Вычислить и вывести на экран в виде таблицы значения функции *F* на интервале от $X_{нач}$ до $X_{кон}$ с шагом *dX*.

$$F(x)=\begin{cases} -a\cdot x^3-b \quad при\ x+c<0\ и\ a\neq 0 \\ \dfrac{x-a}{x+c} \quad при\quad x+c>0\ и\ a=0 \\ \dfrac{x}{c}+\dfrac{c}{x}\ в\ остальных\ случаях \end{cases}$$

где *a, b, c* — действительные числа.

Значения *a, b,* с, $X_{нач}$, $X_{кон}$, *dX* ввести с клавиатуры. Предусмотреть недопущения нестандартных ситуаций.

Вариант 15

Вычислить и вывести на экран в виде таблицы значения функции F на интервале от $X_{нач}$ до $X_{кон}$ с шагом *dX*.

$$F = \begin{cases} -ax^2 + b & при\ x < 0\ и\ b \neq 0 \\ \dfrac{x}{x-c} + 5.5 & при\ x > 0\ и\ b = 0 \\ \dfrac{x}{-c} & в\ остальных\ случаях \end{cases}$$

где a, b, c — действительные числа.

Функция F должна принимать действительное значение, если выражение НЕ(Ац ИЛИ Вц ИЛИ Сц) не равно нулю, и целое значение в противном случае. Через Ац, Вц и Сц обозначены целые части значений a, b, c, операции НЕ и ИЛИ — поразрядные. Значения *a, b, c, $X_{нач}$, $X_{кон}$, dX* ввести с клавиатуры.

Вариант 16

Вычислить и вывести на экран в виде таблицы значения функции *F* на интервале от $X_{нач}$ до $X_{кон}$ с шагом *dX*.

$$F(x) = \begin{cases} a \cdot (x+c)^2 - b & при\ x = 0\ и\ b \neq 0 \\ \dfrac{x-a}{-c} & при\ x = 0\ и\ b = 0 \\ a + \dfrac{x}{c} & в\ остальных\ случаях \end{cases}$$

где *a, b, c* — действительные числа.

Функция F должна принимать действительное значение, если выражение (Ац МОД2 Вц) И НЕ(Ац ИЛИ Сц) не равно нулю, и целое значение в противном случае. Через Ац, Вц и Сц обозначены целые части значений a, b, c, операции НЕ, И, ИЛИ и МОД2 (сложение по модулю 2) — поразрядные. Значения *a, b, c, $X_{нач}$, $X_{кон}$, dX* ввести с клавиатуры.

Вариант 17

Вычислить и вывести на экран в виде таблицы значения функции *F* на интервале от $X_{нач}$ до $X_{кон}$ с шагом *dX*.

$$F(x)=\begin{cases} a\cdot x^2-c\cdot x+b \quad при\ x+10<0\ и\ b\neq 0 \\ \dfrac{x-a}{x-c} \quad при\ \ x+10>0\ и\ b=0 \\ \dfrac{-x}{a-c} \quad в\ остальных\ случаях \end{cases}$$

где *a, b, c* — действительные числа.

Функция F должна принимать действительное значение, если выражение (Ац ИЛИ Вц) И НЕ(Ац ИЛИ Сц) не равно нулю, и целое значение в противном случае. Через Ац, Вц и Сц обозначены целые части значений a, 6, c, операции НЕ, И и ИЛИ — поразрядные. Значения *a, b, c,* $X_{нач}$, $X_{кон}$, *dX* ввести с клавиатуры.

Вариант 18

Вычислить и вывести на экран в виде таблицы значения функции *F* на интервале от $X_{нач}$ до $X_{кон}$ с шагом *dX*.

$$F(x)=\begin{cases} a\cdot x^3+b\cdot x^2 \quad при\ x<0\ и\ b\neq 0 \\ \dfrac{x-a}{x-c} \quad при\ \ x>0\ и\ b=0 \\ \dfrac{x+5}{c\cdot(x-10)} \quad в\ остальных\ случаях \end{cases}$$

где a, b, c — действительные числа.

функция F должна принимать действительное значение, если выражение НЕ(Ац И Вц И Сц) не равно нулю, и целое значение в противном случае. Через Ац, Вц и Сц обозначены целые части значений a, b, c, операции НЕ и И — поразрядные. Значения *a, b, c,* $X_{нач}$, $X_{кон}$, *dX* ввести с клавиатуры.

Вариант 19

Вычислить и вывести на экран в виде таблицы значения функции *F* на интервале от $X_{нач}$ до $X_{кон}$ с шагом *dX*.

$$F(x)=\begin{cases} a\cdot(x+7)^2-b \quad при\ x<5\ и\ b\neq 0 \\ \dfrac{x-c\cdot d}{a\cdot x} \quad при\ \ x>5\ и\ b=0 \\ \dfrac{x}{c}\ в\ остальных\ случаях \end{cases}$$

где a, b, c, d — действительные числа.

Функция F должна принимать действительное значение, если выражение (Ац МОД2 Вц) ИЛИ (Ац МОД2 Сц) не равно нулю, и целое значение в противном случае. Через Ац, Вц и Сц обозначены целые части значений a, b, c, операции ИЛИ и МОД2 (сложение по модулю 2) — поразрядные. Значения *a, b, c, d, $X_{нач}$, $X_{кон}$, dX* ввести с клавиатуры.

Вариант 20

Вычислить и вывести на экран в виде таблицы значения функции *F* на интервале от $X_{нач}$ до $X_{кон}$ с шагом *dX*.

$$F(x)=\begin{cases} -\dfrac{2\cdot x-c}{c\cdot x-a} \quad при\ x<0\ и\ b\neq 0 \\ \dfrac{x-a}{x-c} \quad при\ \ x>0\ и\ b=0 \\ -\dfrac{x}{c}+\dfrac{-c}{2\cdot x}\ в\ остальных\ случаях \end{cases}$$

где *a, b, c* — действительные числа.

Функция F должна принимать действительное значение, если выражение НЕ(Ац ИЛИ Вц) И НЕ(Ац ИЛИ Сц) не равно нулю, и целое значение в противном случае. Через Ац, Вц и Сц обозначены целые части значений a, b, c, операции НЕ, И и ИЛИ — поразрядные. Значения *a, b, c, $X_{нач}$, $X_{кон}$, dX* ввести с клавиатуры.

## Лабораторная работа № 2. Указатели и одномерные массивы

Цель работы – отработка умений и навыков работы с указателями и ссылками; получение навыков работы с одномерными массивами.

### Методические указания

Для выполнения лабораторной работы необходимо изучить следующие разделы курса лекций:

- 1.5.1 Указатели;
- 1.5.2 Ссылки;
- 1.5.3.1 Одномерные массивы.

Указатели предназначены для хранения адресов областей памяти. В C++ различают три вида указателей — указатели на объект, на функцию и на void, отличающиеся свойствами и набором допустимых операций. Указатель не является самостоятельным типом, он всегда связан с каким-либо другим конкретным типом. Величины типа указатель подчиняются общим правилам определения области действия, видимости и времени жизни.

Ссылка представляет собой синоним имени, указанного при инициализации ссылки. Ссылку можно рассматривать как указатель, который всегда разыменовывается. Формат объявления ссылки:

*тип &имя;*

где тип — это тип величины, на которую указывает ссылка, & — оператор ссылки, означающий, что следующее за ним имя является именем переменной ссылочного типа. Ссылка, в отличие от указателя, не занимает дополнительного пространства в памяти и является просто другим именем величины. Операция над ссылкой приводит к изменению величины, на которую она ссылается.

Конечная именованная последовательность однотипных величин называется массивом. Описание массива в программе отличается от описания простой переменной наличием после имени квадратных скобок, в которых задается количество элементов массива (размерность):

*float a[10];  // описание массива из 10 вещественных чисел*

Элементы массива нумеруются с нуля.

## Пример выполнения лабораторной работы

*Задание:*

В одномерном массиве, состоящем из *n* вещественных элементов, вычислить:

- сумму отрицательных элементов массива;
- произведение элементов массива, расположенных между максимальным и минимальным элементами.

Упорядочить элементы массива по возрастанию. Размер и элементы массива задавать с клавиатуры.

*Листинг программы:*

```
#include <iostream>
using namespace std;

int main()
{
	setlocale(0, "rus");
	int n;
	cout<<"Введите количество элементов: ";
	cin >> n;
	int *a = new int [n];
	int sum = 0;
	int num = 0;
	cout<<"Введите элементы: \n";
	for(int i = 0; i < n; i++)
	{
		cin >> a[i];
		if(a[i] < 0)
		{
			sum += a[i];
			num++;
		}
	}
	if(num == 0)
		cout<<"Нет отрицательных элементов!\n";
	else
		cout<<"Сумма отрицательных элементов: "<<sum;
	int temp;
	for(int i = 0; i < n; i++)
	{
		for(int j = 0; j < n-1; j++)
		{
			if(a[j] > a[j+1])
			{
				temp = a[j];
```

```
                        a[j] = a[j+1];
                        a[j+1] = temp;
                }
        }
    }
    cout<<"\nОтсортированный массив:\n";
    for(int i = 0; i < n; i++)
        cout<<a[i]<<" ";
    int max = a[0];
    int min = a[0];
    int im, jm = 0;
    for(int i = 0; i < n; i++)
    {
        if(a[i] > max)
        {
            max = a[i];
            im = i;
        }
        if(a[i] < min)
        {
            min = a[i];
            jm = i;
        }
    }
    int proz = 1;
    if(im > jm)
    {
        for(int i = jm; i <= im ; i++)
        {
            proz *= a[i];
        }
    }
    else if(jm > im)
    {
        for(int i = jm; i <= im ; i++)
        {
            proz *= a[i];
        }
    }
    cout<<"\n\nПроизведение элементов массива, расположенных между максимальным и
минимальным элементами: "<<proz;
    _gettch();
}
```

## Варианты заданий

Вариант 1

В одномерном массиве, состоящем из *n* вещественных элементов, вычислить:

- сумму отрицательных элементов массива;
- произведение элементов массива, расположенных между максимальным и минимальным элементами.

Упорядочить элементы массива по возрастанию. Размер и элементы массива задавать с клавиатуры.

Вариант 2

В одномерном массиве, состоящем из *n* вещественных элементов, вычислить:

- сумму положительных элементов массива;
- произведение элементов массива, расположенных между максимальным по модулю и минимальным по модулю элементами.

Упорядочить элементы массива по убыванию. Размер и элементы массива задавать с клавиатуры.

Вариант 3

В одномерном массиве, состоящем из ***n*** целых элементов, вычислить:

- произведение элементов массива с четными номерами;
- сумму элементов массива, расположенных между первым и последним нулевыми элементами.

Преобразовать массив таким образом, чтобы сначала располагались все положительные элементы, а потом — все отрицательные (элементы, равные 0, считать положительными). Размер и элементы массива задавать с клавиатуры.

Вариант 4

В одномерном массиве, состоящем из *n* вещественных элементов, вычислить:

- сумму элементов массива с нечетными номерами;
- сумму элементов массива, расположенных между первым и последним отрицательными элементами.

Сжать массив, удалив из него все элементы, модуль которых не превышает 1. Освободившиеся в конце массива элементы заполнить нулями. Размер и элементы массива задавать с клавиатуры.

Вариант 5

В одномерном массиве, состоящем из ***n*** вещественных элементов, вычислить:

- максимальный элемент массива;
- сумму элементов массива, расположенных до последнего положительного элемента.

Сжать массив, удалив из него все элементы, модуль которых находится в интервале [a,b]. Освободившиеся в конце массива элементы заполнить нулями. a и b, размер и элементы массива задавать с клавиатуры.

Вариант 6

В одномерном массиве, состоящем из *n* вещественных элементов, вычислить:

- минимальный элемент массива;
- сумму элементов массива, расположенных между первым и последним положительными элементами.

Преобразовать массив таким образом, чтобы сначала располагались все элементы, равные нулю, а потом — все остальные. Размер и элементы массива задавать с клавиатуры.

Вариант 7

В одномерном массиве, состоящем из *n* целых элементов, вычислить:

- номер максимального элемента массива;
- произведение элементов массива, расположенных между первым и вторым нулевыми элементами.

Преобразовать массив таким образом, чтобы в первой его половине располагались элементы, стоявшие в нечетных позициях, а во второй половине — элементы, стоявшие в четных позициях. Размер и элементы массива задавать с клавиатуры.

Вариант 8

В одномерном массиве, состоящем из *n* вещественных элементов, вычислить:

- номер минимального элемента массива;
- сумму элементов массива, расположенных между первым и вторым отрицательными элементами.

Преобразовать массив таким образом, чтобы сначала располагались все элементы, модуль которых не превышает 1, а потом — все остальные. Размер и элементы массива задавать с клавиатуры.

Вариант 9

В одномерном массиве, состоящем из *n* вещественных элементов, вычислить:

- максимальный по модулю элемент массива;
- сумму элементов массива, расположенных между первым и вторым положительными элементами.

Преобразовать массив таким образом, чтобы элементы, равные нулю, располагались после всех остальных. Размер и элементы массива задавать с клавиатуры.

Вариант 10

В одномерном массиве, состоящем из *n* целых элементов, вычислить:

- минимальный по модулю элемент массива;

- сумму модулей элементов массива, расположенных после первого элемента, равного нулю.

Преобразовать массив таким образом, чтобы в первой его половине располагались элементы, стоявшие в четных позициях, а во второй половине — элементы, стоявшие в нечетных позициях. Размер и элементы массива задавать с клавиатуры.

## Вариант 11

В одномерном массиве, состоящем из *n* вещественных элементов, вычислить:

- номер минимального по модулю элемента массива;
- сумму модулей элементов массива, расположенных после первого отрицательного элемента.

Сжать массив, удалив из него все элементы, величина которых находится в интервале [a,b]. Освободившиеся в конце массива элементы заполнить нулями. Размер и элементы массива задавать с клавиатуры.

## Вариант 12

В одномерном массиве, состоящем из *n* вещественных элементов, вычислить:

- номер максимального по модулю элемента массива;
- сумму элементов массива, расположенных после первого положительного элемента.

Преобразовать массив таким образом, чтобы сначала располагались все элементы, целая часть которых лежит в интервале [a,b], а потом — все остальные. Размер и элементы массива задавать с клавиатуры.

Вариант 13

В одномерном массиве, состоящем из *n* вещественных элементов, вычислить:

- количество элементов массива, лежащих в диапазоне от A до B;
- сумму элементов массива, расположенных после максимального элемента.

Упорядочить элементы массива по убыванию модулей элементов. Размер и элементы массива задавать с клавиатуры.

Вариант 14

В одномерном массиве, состоящем из *n* вещественных элементов, вычислить:

- количество элементов массива, равных 0;
- сумму элементов массива, расположенных после минимального элемента.

Упорядочить элементы массива по возрастанию модулей элементов. Размер и элементы массива задавать с клавиатуры.

Вариант 15

В одномерном массиве, состоящем из *n* вещественных элементов, вычислить:

- количество элементов массива, больших C;
- произведение элементов массива, расположенных после максимального по модулю элемента.

Преобразовать массив таким образом, чтобы сначала располагались все отрицательные элементы, а потом — все положительные (элементы, равные 0, считать положительными). Размер и элементы массива задавать с клавиатуры.

Вариант 16

В одномерном массиве, состоящем из *n* вещественных элементов, вычислить:

- количество отрицательных элементов массива;
- сумму модулей элементов массива, расположенных после минимального по модулю элемента.

Заменить все отрицательные элементы массива их квадратами и упорядочить элементы массива по возрастанию. Размер и элементы массива задавать с клавиатуры.

Вариант 17

В одномерном массиве, состоящем из *n* целых элементов, вычислить:

- количество положительных элементов массива;
- сумму элементов массива, расположенных после последнего элемента, равного нулю.

Преобразовать массив таким образом, чтобы сначала располагались все элементы, целая часть которых не превышает 1, а потом — все остальные. Размер и элементы массива задавать с клавиатуры.

Вариант 18

В одномерном массиве, состоящем из *n* вещественных элементов, вычислить:

- количество элементов массива, меньших C;
- сумму целых частей элементов массива, расположенных после последнего отрицательного элемента.

Преобразовать массив таким образом, чтобы сначала располагались все элементы, отличающиеся от максимального не более чем на 20%, а потом — все остальные. Размер и элементы массива задавать с клавиатуры.

Вариант 19

В одномерном массиве, состоящем из *n* вещественных элементов, вычислить:

- произведение отрицательных элементов массива;
- сумму положительных элементов массива, расположенных до максимального элемента.

Изменить порядок следования элементов в массиве на обратный. Размер и элементы массива задавать с клавиатуры.

Вариант 20

В одномерном массиве, состоящем из *n* вещественных элементов» вычислить:

- произведение положительных элементов массива;
- сумму элементов массива, расположенных до минимального элемента.

Упорядочить по возрастанию отдельно элементы, стоящие на четных местах, и элементы, стоящие на нечетных местах.

Вариант 21

В одномерном массиве, состоящем из *n* целых элементов, вычислить:

- количество элементов массива, не превышающих заданного числа N;
- произведение элементов массива, расположенных после последнего элемента, равного нулю.

Преобразовать массив таким образом, чтобы сначала располагались все элементы, модуль которых не превышает 1, а потом — все остальные. Размер и элементы массива задавать с клавиатуры.

Вариант 22

В одномерном массиве, состоящем из *n* вещественных элементов, вычислить:

- количество элементов массива, больших C1, но меньших C2, причем (C2>C1);
- сумму целых частей элементов массива, расположенных после последнего отрицательного элемента.

Преобразовать массив таким образом, чтобы сначала располагались все элементы, отличающиеся от максимального не более чем на 30%, а потом — все остальные.  Размер и элементы массива задавать с клавиатуры.

## Лабораторная работа № 3. Многомерные массивы

Цель работы – получение навыков работы с многомерными массивами в C++.

### Методические указания

Для выполнения лабораторной работы необходимо изучить следующие разделы курса лекций:

- 1.5.3.2 Динамические массивы;
- 1.5.3.3 Многомерные массивы;

Динамические массивы создают с помощью операции *new*, при этом необходимо указать тип и размерность. Преимущество динамических массивов состоит в том, что размерность может быть переменной, то есть объем памяти, выделяемой под массив, определяется па этане выполнения программы. Доступ к элементам динамического массива осуществляется точно так же, как к элементам статического массива.

Многомерные массивы задаются указанием каждого измерения в квадратных скобках, например, оператор

*int matr [6][8];*

задает описание двумерного массива из 6 строк и 8 столбцов. В памяти такой массив располагается в последовательных ячейках построчно. Многомерные массивы размещаются так, что при переходе к следующему элементу быстрее всего изменяется последний индекс.

Для создания динамического многомерного массива необходимо указать в операции *new* все его размерности (самая левая размерность может быть переменной).

### Пример выполнения лабораторной работы

*Задание:*

Дана целочисленная прямоугольная матрица. Определить:

- количество строк, не содержащих ни одного нулевого элемента;

- максимальное из чисел, встречающихся в заданной матрице более одного раза.

Размер и элементы матрицы задавать с клавиатуры. Оформить каждый пункт задания в виде отдельной функции и организовать правильный вызов функций.

*Листинг программы:*

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

bool Find(int val, int* ar, int size, int pos = 0)
{
	for(int i = pos; i < size; i++)
		if(ar[i] == val)
			return true;
	return false;
}

int main()
{
	setlocale(0, "rus");
	int n, m, **matr;
	cout<<"n = ";
	cin>>n;
	cout<<"m = ";
	cin>>m;
	matr=new int*[n];
	for(int i=0; i<n; i++)
	{
		matr[i]=new int[m];
		for(int j=0; j<m; j++)
		{
			cout<<"["<<i<<"]["<<j<<"] = ";
			cin>>matr[i][j];
		}
	}
	int  count   = 0;
	int  max_val = INT_MIN;
	for(int i = 0; i < n; i++)
	{
		count += (int)!Find(0, matr[i], n);
		for(int j = 0; j < n; j++)
			if(matr[i][j] > max_val)
				max_val = matr[i][j];
	}
	cout << "Количество строк, не содержащих нулей: " << count << endl;
	cout << "Максимальный повторяющийся элемент: " << max_val << endl;
	_gettch();
	return 0;
}
```

## Варианты заданий

Вариант 1

Дана целочисленная прямоугольная матрица. Определить:

- количество строк, не содержащих ни одного нулевого элемента;
- максимальное из чисел, встречающихся в заданной матрице более одного раза.

Размер и элементы матрицы задавать с клавиатуры. Оформить каждый пункт задания в виде отдельной функции и организовать правильный вызов функций.

Вариант 2

Дана целочисленная прямоугольная матрица. Определить количество столбцов, не содержащих ни одного нулевого элемента. Характеристикой строки целочисленной матрицы назовем сумму ее положительных четных элементов.

Переставляя строки заданной матрицы, расположить их в соответствии с ростом характеристик.

Размер и элементы матрицы задавать с клавиатуры.

Оформить каждый пункт задания в виде отдельной функции и организовать правильный вызов функций.

Вариант 3

Дана целочисленная прямоугольная матрица. Определить;

- количество столбцов, содержащих хотя бы один нулевой элемент;
- номер строки, в которой находится самая длинная серия одинаковых элементов.

Размер и элементы матрицы задавать с клавиатуры.

Оформить каждый пункт задания в виде отдельной функции и организовать правильный вызов функций.

Вариант 4

Дана целочисленная квадратная матрица. Определить:

- произведение элементов в тех строках, которые не содержат отрицательных элементов;
- максимум среди сумм элементов диагоналей, параллельных главной диагонали матрицы.

Размер и элементы матрицы задавать с клавиатуры.

Оформить каждый пункт задания в виде отдельной функции и организовать правильный вызов функций.

Вариант 5

Дана целочисленная квадратная матрица. Определить:

- сумму элементов в тех столбцах, которые не содержат отрицательных элементов;
- минимум среди сумм модулей элементов диагоналей, параллельных побочной диагонали матрицы.

Размер и элементы матрицы задавать с клавиатуры.

Оформить каждый пункт задания в виде отдельной функции и организовать правильный вызов функций.

Вариант 6

Дана целочисленная прямоугольная матрица. Определить:

- сумму элементов в тех строках, которые содержат хотя бы один отрицательный элемент;
- номера строк и столбцов всех седловых точек матрицы.

Матрица A имеет седловую точку $A_{ij}$, если $A_{ij}$ является минимальным элементом в $i$-й строке и максимальным в $j$-м столбце.

Размер и элементы матрицы задавать с клавиатуры.

Оформить каждый пункт задания в виде отдельной функции и организовать правильный вызов функций.

Вариант 7

Для заданной матрицы размером 8 на 8 найти такие *k*, что *k-я* строка матрицы совладает с *k-м* столбцом.

Найти сумму элементов в тех строках, которые содержат хотя бы один отрицательный элемент. Размер и элементы матрицы задавать с клавиатуры.

Оформить каждый пункт задания в виде отдельной функции и организовать правильный вызов функций.

Вариант 8

Характеристикой столбца целочисленной матрицы назовем сумму модулей его отрицательных нечетных элементов.

Переставляя столбцы заданной матрицы, расположить их в соответствии с ростом характеристик.

Найти сумму элементов в тех столбцах, которые содержат хотя бы один отрицательный элемент.

Размер и элементы матрицы задавать с клавиатуры. Найти сумму элементов в тех строках, которые содержат хотя бы один отрицательный элемент. Размер и элементы матрицы задавать с клавиатуры.

Оформить каждый пункт задания в виде отдельной функции и организовать правильный вызов функций.

Вариант 9

Соседями элемента $A_y$ в матрице назовем элементы $A_к$; $i-1 \le k \le i+1,\ j-1 \le l \le j+1, (k,l) \ne (i,j)$. Операция сглаживания матрицы дает новую матрицу того же размера, каждый элемент которой получается как среднее арифметическое имеющихся соседей соответствующего элемента исходной

матрицы. Построить результат сглаживания заданной вещественной матрицы размером 10 на 10. В сглаженной матрице найти сумму модулей элементов, расположенных ниже главной диагонали. Элементы матрицы задавать с клавиатуры.

Оформить каждый пункт задания в виде отдельной функции и организовать правильный вызов функций.

Вариант 10

Элемент матрицы называется локальным минимумом, если он строго меньше всех имеющихся у него соседей.

Подсчитать количество локальных минимумов заданной матрицы размером 10 на 10.

Найти сумму модулей элементов, расположенных выше главной диагонали.

Элементы матрицы задавать с клавиатуры. Найти сумму элементов в тех строках, которые содержат хотя бы один отрицательный элемент. Размер и элементы матрицы задавать с клавиатуры. Оформить каждый пункт задания в виде отдельной функции и организовать правильный вызов функций.

Вариант 11

Коэффициенты системы линейных уравнений заданы в виде прямоугольной матрицы.

С помощью допустимых преобразований привести систему к треугольному виду.

Найти количество строк, среднее арифметическое элементов которых меньше заданной величины.

Размер и элементы матрицы задавать с клавиатуры. Оформить каждый пункт задания в виде отдельной функции и организовать правильный вызов функций.

Вариант 12

Уплотнить заданную матрицу, удаляя из нее строки и столбцы, заполненные нулями. Найти номер первой из строк, содержащих хотя бы один положительный элемент. Размер и элементы матрицы задавать с клавиатуры.

Оформить каждый пункт задания в виде отдельной функции и организовать правильный вызов функций.

Вариант 13

Осуществить циклический сдвиг элементов прямоугольной матрицы на *n* элементов вправо или вниз (в зависимости от введенного режима), *n* должно быть не больше количества элементов в строке или столбце. Размер и элементы матрицы задавать с клавиатуры.

Оформить каждый пункт задания в виде отдельной функции и организовать правильный вызов функций.

Вариант 14

Осуществить циклический сдвиг элементов квадратной матрицы размерности *MxN* вправо на *k* элементов таким образом: элементы 1-й строки сдвигаются в последний столбец сверху вниз, из него — в последнюю строку справа налево, из нее — в первый столбец снизу вверх, из него — в первую строку; для остальных элементов — аналогично. Размер и элементы матрицы задавать с клавиатуры.

Оформить каждый пункт задания в виде отдельной функции и организовать правильный вызов функций.

Вариант 15

Дана целочисленная прямоугольная матрица. Определить номер первого из столбцов, содержащих хотя бы один нулевой элемент. Характеристикой строки целочисленной матрицы назовем сумму ее отрицательных четных элементов.

Переставляя строки заданной матрицы, расположить их в соответствии с убыванием характеристик. Размер и элементы матрицы задавать с клавиатуры.

Оформить каждый пункт задания в виде отдельной функции и организовать правильный вызов функций.

Вариант 16

Упорядочить строки целочисленной прямоугольной матрицы по возрастанию количества одинаковых элементов в каждой строке. Найти номер первого из столбцов, не содержащих ни одного отрицательного элемента. Размер и элементы матрицы задавать с клавиатуры.

Оформить каждый пункт задания в виде отдельной функции и организовать правильный вызов функций.

Вариант 17

Путем перестановки элементов квадратной вещественной матрицы добиться того, чтобы ее максимальный элемент находился в левом верхнем углу, следующий по величине — в позиции (2,2), следующий по величине — в позиции (3,3) и т. д., заполнив таким образом всю главную диагональ.

Найти номер первой из строк, не содержащих ни одного положительного элемента. Размер и элементы матрицы задавать с клавиатуры.

Оформить каждый пункт задания в виде отдельной функции и организовать правильный вызов функций.

Вариант 18

Дана целочисленная прямоугольная матрица. Определить:

- количество строк, содержащих хотя бы один нулевой элемент;
- номер столбца, в которой находится самая длинная серия одинаковых элементов.

Размер и элементы матрицы задавать с клавиатуры.

Оформить каждый пункт задания в виде отдельной функции и организовать правильный вызов функций.

### Вариант 19

Дана целочисленная квадратная матрица. Определить:

- сумму элементов в тех строках, которые не содержат отрицательных элементов;
- минимум среди сумм элементов диагоналей, параллельных главной диагонали матрицы.

Размер и элементы матрицы задавать с клавиатуры.

Оформить каждый пункт задания в виде отдельной функции и организовать правильный вызов функций.

### Вариант 20

Дана целочисленная прямоугольная матрица. Определить:

- количество отрицательных элементов в тех строках, которые содержат хотя бы один нулевой элемент;
- номера строк и столбцов всех седловых точек матрицы.

Матрица $A$ имеет седловую точку $A_{ij}$, если $A_{ij}$ является минимальным элементом в *i-й* строке и максимальным в *j*-м столбце.

Оформить каждый пункт задания в виде отдельной функции и организовать правильный вызов функций.

## Лабораторная работа № 4. Структуры

Цель работы – получение навыков работы по созданию типов данных, определяемых пользователем.

### Методические указания

Для выполнения лабораторной работы необходимо изучить следующие разделы курса лекций:

- 1.6 Типы данных, определяемые пользователем.

В отличие от массива, все элементы которого однотипны, структура может содержать элементы разных типов. В языке C++ структура является видом класса и обладает всеми его свойствами, но во многих случаях достаточно использовать структуры так, как они определены в языке C.

```
struct [ имя_типа ]
{
        тип_1 элемент_1;
        тип 2 элемент 2;
        ……….
        тип_n эленент_n;
} [ список_описателей ];
```

### Пример выполнения лабораторной работы

*Задание:*

1. Описать структуру с именем STUDENT, содержащую следующие поля:
   - NAME — фамилия и инициалы;
   - номер группы;

     успеваемость (массив из пяти элементов).
2. Написать программу, выполняющую следующие действия:

- ввод с клавиатуры данных в массив, состоящий из десяти структур типа STUDENT; записи должны быть упорядочены по возрастанию номера группы;
- вывод на дисплей фамилий и номеров групп для всех студентов, включенных в массив, если средний балл студента больше 4,0;

Если таких студентов нет, вывести соответствующее сообщение.

*Листинг программы:*

```
#include <iostream>
#include <stdlib.h>
#include <stdio.h>

#define MAX_LENGTH_NAME     32
#define NDISCIPLINES         5
#define PROGRESS_TRESHOLD   4.0

typedef struct student_tag {
	char name[MAX_LENGTH_NAME];
	int group;
	int marks[NDISCIPLINES];
} student_t;

#define NSTUDENTS 10

int main()
{
	setlocale(0, "rus");
	int i;
	int j;
	float avg;
	student_t tmp;
	student_t students[NSTUDENTS];
	for(i = 0; i < NSTUDENTS; ++i) {
		printf("Введите фамилию и инициалы: ");
		gets_s(students[i].name);
		printf("Введите номер группы: ");
		if(scanf_s("%d", &students[i].group) != 1) {
			fprintf(stderr, "data reading error\n");
			return EXIT_FAILURE;
		}
		printf("Введите оценки:\n");
		for(j = 0; j < NDISCIPLINES; ++j) {
			if(scanf_s("%d", &students[i].marks[j]) != 1) {
				fprintf(stderr, "data reading error\n");
				return EXIT_FAILURE;
			}
		}
		fflush(stdin);
	}
	for(i = 0; i < NSTUDENTS; ++i) {
		for(j = NSTUDENTS - 1; j >= i; --j) {
			if(students[j].group < students[j-1].group) {
				tmp = students[j];
				students[j] = students[j-1];
				students[j-1] = tmp;
```

```
                        }
                }
        }
        printf("\nСписок студентов, средний бал у которых выше %f:\n",
PROGRESS_TRESHOLD);
        int number = 0;
        for(i = 0; i < NSTUDENTS; ++i)
        {
                avg = 0;
                for(j = 0; j < NDISCIPLINES; ++j)
                        avg += students[i].marks[j];
                avg /= NDISCIPLINES;
                if(avg > PROGRESS_TRESHOLD)
                {
                        number++;
                        printf("ФИО: %s\nГРУППА: %d\nОЦЕНКИ: ", students[i].name,
students[i].group);
                        for(j = 0; j < NDISCIPLINES; ++j)
                                printf("%d, ", students[i].marks[j]);
                        printf("\n\n");
                }
        }
        if(number==0)
                printf("Таких студентов нет!");
        _gettch();
        return 0;
}
```

## Варианты заданий

Вариант 1

1. Описать структуру с именем STUDENT, содержащую следующие поля:

- NAME — фамилия и инициалы;
- номер группы;

  успеваемость (массив из пяти элементов).

2. Написать программу, выполняющую следующие действия:

- ввод с клавиатуры данных в массив, состоящий из десяти структур типа STUDENT; записи должны быть упорядочены по возрастанию номера группы;
- вывод на дисплей фамилий и номеров групп для всех студентов, включенных в массив, если средний балл студента больше 4,0;

Если таких студентов нет, вывести соответствующее сообщение.

Вариант 2

1. Описать структуру с именем STUDENT, содержащую следующие поля:
   - фамилия и инициалы;
   - номер группы;
   - успеваемость (массив из пяти элементов).
2. Написать программу, выполняющую следующие действия:
   - ввод с клавиатуры данных в массив, состоящий из десяти структур типа STUDENT; записи должны быть упорядочены по возрастанию среднего балла;
   - вывод на дисплей фамилий и номеров групп для всех студентов, имеющих оценки 4 и 5;

   Если таких студентов нет, вывести соответствующее сообщение.

Вариант 3

1. Описать структуру с именем STUDENT, содержащую следующие поля:
   - фамилия и инициалы;
   - номер группы;
   - успеваемость (массив из пяти элементов).
2. Написать программу, выполняющую следующие действия:
   - ввод с клавиатуры данных в массив, состоящий из десяти структур типа STUDENT; записи должны быть упорядочены по алфавиту;
   - вывод на дисплей фамилий и номеров групп для всех студентов, имеющих хотя бы одну оценку 2;

Если таких студентов нет, вывести соответствующее сообщение.

Вариант 4

1. Описать структуру с именем AEROFLOT, содержащую следующие поля:

- название пункта назначения рейса;
- номер рейса;
- тип самолета.

2. Написать программу, выполняющую следующие действия:

- ввод с клавиатуры данных в массив, состоящий из семи элементов типа AEROFLOT; записи должны быть упорядочены по возрастанию номера рейса;
- вывод на экран номеров рейсов и типов самолетов, вылетающих в пункт назначения, название которого совпало с названием, введенным с клавиатуры;

Если таких рейсов нет, выдать на дисплей соответствующее сообщение.

Вариант 5

1. Описать структуру с именем AEROFLOT, содержащую следующие поля:

- название пункта назначения рейса;
- номер рейса;
- тип самолета.

2. Написать программу, выполняющую следующие действия:

- ввод с клавиатуры данных в массив, состоящий из семи элементов типа AEROFLOT; записи должны быть размещены в алфавитном порядке по названиям пунктов назначения;
- вывод на экран пунктов назначения и номеров рейсов, обслуживаемых самолетом, тип которого введен с клавиатуры;

Если таких рейсов нет, выдать на дисплей соответствующее сообщение.

Вариант 6

1. Описать структуру с именем WORKER, содержащую следующие поля:

- фамилия и инициалы работника;
- название занимаемой должности;
- год поступления на работу.

2. Написать программу, выполняющую следующие действия:

- ввод с клавиатуры данных в массив, состоящий из десяти структур типа WORKER; записи должны быть размещены по алфавиту.
- вывод на дисплей фамилий работников, чей стаж работы в организации превышает значение, введенное с клавиатуры;

Если таких работников нет, вывести на дисплей соответствующее сообщение.

Вариант 7

1. Описать структуру с именем TRAIN, содержащую следующие поля:

- название пункта назначения;
- номер поезда;
- время отправления.

2. Написать программу, выполняющую следующие действия:

- ввод с клавиатуры данных в массив, состоящий из восьми элементов типа TRAIN; записи должны быть размещены в алфавитном порядке по названиям пунктов назначения;
- вывод на экран информации о поездах, отправляющихся после введенного с клавиатуры времени;

Если таких поездов нет, выдать па дисплей соответствующее сообщение.

Вариант 8

1. Описать структуру с именем TRAIN, содержащую следующие поля:

- название пункта назначения;
- номер поезда;

- время отправления.

2. Написать программу, выполняющую следующие действия:

- ввод с клавиатуры данных в массив, состоящий из шести элементов типа TRAIN; записи должны быть упорядочены по времени отправления поезда;
- вывод на экран информации о поездах, направляющихся в пункт, название которого введено с клавиатуры;

Если таких поездов нет, выдать на дисплей соответствующее сообщение.

Вариант 9

1. Описать структуру с именем TRAIN, содержащую следующие поля:

- название пункта назначения;
- номер поезда;
- время отправления.

2. Написать программу, выполняющую следующие действия:

- ввод с клавиатуры данных в массив, состоящий из восьми элементов типа TRAIN; записи должны быть упорядочены по номерам поездов;
- вывод на экран информации о поезде, номер которого введен с клавиатуры;

Если таких поездов нет, выдать на дисплей соответствующее сообщение.

Вариант 10

1. Описать структуру с именем MARSH, содержащую следующие поля:

- название начального пункта маршрута;
- название конечного пункта маршрута;
- номер маршрута.

2. Написать программу, выполняющую следующие действия:

- ввод с клавиатуры данных в массив, состоящий из восьми элементов типа MARSH; записи должны быть упорядочены по номерам маршрутов;
- вывод на экран информации о маршруте, номер которого введен с клавиатуры;

Если таких маршрутов нет, выдать па дисплей соответствующее сообщение.

Вариант 11

1. Описать структуру с именем MARSH, содержащую следующие поля:
   - название начального пункта маршрута;
   - название конечного пункта маршрута;
   - номер маршрута.
2. Написать программу, выполняющую следующие действия:
   - ввод с клавиатуры данных в массив, состоящий из восьми элементов типа MARSH; записи должны быть упорядочены но номерам маршрутов;
   - вывод на экран информации о маршрутах, которые начинаются или кончаются в пункте, название которого введено с клавиатуры;

Если таких маршрутов нет, выдать па дисплей соответствующее сообщение.

Вариант 12

1. Описать структуру с именем NOTE, содержащую следующие поля:
   - фамилия, имя;
   - номер телефона;
   - день рождения (массив из трех чисел).
2. Написать программу, выполняющую следующие действия:

- ввод с клавиатуры данных в массив, состоящий из восьми элементов типа NOTE; записи должны быть упорядочены по датам дней рождения;
- вывод па экран информации о человеке, номер телефона которого введен с клавиатуры;

Если такого нет, выдать на дисплей соответствующее сообщение.

Вариант 13

1. Описать структуру с именем NOTE, содержащую следующие поля:
   - фамилия, имя;
   - номер телефона;
   - день рождения (массив из трех чисел).
2. Написать программу, выполняющую следующие действия:
   - ввод с клавиатуры данных в массив, состоящий из восьми элементов типа NOTE; записи должны быть размещены по алфавиту;
   - вывод на экран информации о людях, чьи дни рождения приходятся на месяц, значение которого введено с клавиатуры;

Если таких нет, выдать на дисплей соответствующее сообщение.

Вариант 14

1. Описать структуру с именем NOTE, содержащую следующие поля:
   - фамилия, имя;
   - номер телефона;
   - день рождения (массив из трех чисел).
2. Написать программу, выполняющую следующие действия:
   - ввод с клавиатуры данных в массив, состоящий из восьми элементов типа NOTE; записи должны быть упорядочены по трем первым цифрам номера телефона;

- вывод на экран информации о человеке, чья фамилия введена с клавиатуры;

Если такого нет, выдать па дисплей соответствующее сообщение.

Вариант 15

1. Описать структуру с именем ZNAK, содержащую следующие поля:
   - фамилия, имя;
   - знак Зодиака;
   - день рождения (массив из трех чисел).
2. Написать программу, выполняющую следующие действия:
   - ввод с клавиатуры данных в массив, состоящий из восьми элементов типа ZNAK; записи должны быть упорядочены по датам дней рождения;
   - вывод на экран информации о человеке, чья фамилия введена с клавиатуры;

Если такого нет, выдать па дисплей соответствующее сообщение.

Вариант 16

1. Описать структуру с именем ZNAK, содержащую следующие поля:
   - фамилия, имя;
   - знак Зодиака;
   - день рождения (массив из трех чисел).
2. Написать программу, выполняющую следующие действия:
   - ввод с клавиатуры данных в массив, состоящий из восьми элементов типа ZNAK; записи должны быть упорядочены по датам дней рождения;

- вывод на экран информации о людях, родившихся под знаком, наименование которого введено с клавиатуры;

Если таких нет, выдать на дисплей соответствующее сообщение.

Вариант 17

1. Описать структуру с именем ZNAK, содержащую следующие поля:
   - фамилия, имя;
   - знак Зодиака;
   - день рождения (массив из трех чисел).
2. Написать программу, выполняющую следующие действия:
   - ввод с клавиатуры данных в массив, состоящий из восьми элементов типа ZNAK; записи должны быть упорядочены по знакам Зодиака;
   - вывод па экран информации о людях, родившихся в месяц, значение которого введено с клавиатуры;

Если таких нет, выдать на дисплей соответствующее сообщение.

Вариант 18

1. Описать структуру с именем PRICE, содержащую следующие поля:
   - название товара;
   - название магазина, в котором продается товар;
   - стоимость товара в руб.
2. Написать программу, выполняющую следующие действия:
   - ввод с клавиатуры данных в массив, состоящий из восьми элементов типа PRICE; записи должны быть размещены в алфавитном порядке по названиям товаров;
   - вывод на экран информации о товаре, название которого введено с клавиатуры;

Если таких товаров нет, выдать на дисплей соответствующее сообщение.

Вариант 19

1. Описать структуру с именем PRICE, содержащую следующие поля:
   - название товара;
   - название магазина, в котором продается товар;
   - стоимость товара в руб.
2. Написать программу, выполняющую следующие действия:
   - ввод с клавиатуры данных в массив, состоящий из восьми элементов типа PRICE; записи должны быть размещены в алфавитном порядке по названиям магазинов;
   - вывод на экран информации о товарах, продающихся в магазине, название которого введено с клавиатуры;

Если такого магазина нет, выдать на дисплей соответствующее сообщение.

Вариант 20

1. Описать структуру с именем ORDER, содержащую следующие поля:
   - расчетный счет плательщика;
   - расчетный счет получателя;
   - перечисляемая сумма в руб.
2. Написать программу, выполняющую следующие действия:
   - ввод с клавиатуры данных в массив, состоящий из восьми элементов типа ORDER; записи должны быть размещены в алфавитном порядке по расчетным счетам плательщиков;
   - вывод на экран информации о сумме, снятой с расчетного счета плательщика, введенного с клавиатуры;

Если такого расчетного счета нет, выдать на дисплей соответствующее сообщение.

## Лабораторная работа № 5. Простейшие функции

Цель работы – отработка умений, навыков создания и работы с простейшими функциями в C++.

### Методические указания

Для выполнения лабораторной работы необходимо изучить следующие разделы курса лекций:

- 2.1 Функции;
- 2.2 Директивы препроцессора;
- 2.3 Области действия идентификаторов.

Функция — это именованная последовательность описаний и операторов, выполняющая какое-либо законченное действие. Функция может принимать параметры и возвращать значение.

Любая программа на C++ состоит из функций, одна из которых должна иметь имя *main* (с нее начинается выполнение программы). Функция начинает выполняться в момент вызова. Любая функция должна быть объявлена и определена. Как и для других величин, объявлений может быть несколько, а определение только одно. Объявление функции должно находиться в тексте раньше ее вызова для того, чтобы компилятор мог осуществить проверку правильности вызова.

Препроцессором называется первая фаза компилятора. Инструкции препроцессора называются директивами. Они должны начинаться с символа #, перед которым в строке могут находиться только пробельные символы.

Директива *#include <имя_файла>* вставляет содержимое указанного файла в ту точку исходного файла, где она записана. Включаемый файл также может содержать директивы *#include*. Поиск файла, если не указан полный путь, ведется в стандартных каталогах включаемых файлов. Вместо угловых скобок могут использоваться кавычки (" ") — в этом случае поиск файла ведется в каталоге, содержащем исходный файл, а затем уже в стандартных каталогах.

Директива #define определяет подстановку в тексте программы. Она используется для определения *символических констант*, *макросов*, *символов*, *управляющих условной компиляцией*.

Каждый программный объект имеет область действия, которая определяется видом и местом его объявления. Существуют следующие области действия: блок, файл, функция, прототип функции, класс и поименованная область.

## Пример выполнения лабораторной работы

*Задание:*

Для хранения данных о цветных планшетных сканерах описать структуру вида:

```
struct scan_info
{
char model[25]:    // наименование подели
int price;    // цена
double x_size;      // горизонтальный размер области сканирования
double y_size;      // вертикальный размер области сканирования
int optr;     // оптическое разрешение
int grey:     // число градаций серого
};
```

Написать функцию, которая записывает в бинарный файл данные о сканере из приведенной структуры. Структура файла: в первых двух байтах размещается значение типа int, определяющее количество сделанных в файл записей; далее без пропусков размещаются записи о сканерах.

*Листинг программы:*

```
#include <iostream>
#include <fstream>
using namespace std;
struct scan_info{
       char model[25];
       int price;
```

```
	double x_size;
	double y_size;
	int optr;
	int grey;
};
void safe( fstream & out, const scan_info & data );
int load( fstream & in, scan_info & data, size_t pos );
int main()
{
	setlocale(0, "rus");
	char * filename = "data.dat";
	fstream file( filename, ios::out | ios::in | ios::binary );
	if( !file.is_open())
	{
		ofstream file2( filename, ios::binary );
		wchar_t s = 0;
		file2.write(reinterpret_cast<char*>(&s), sizeof(wchar_t) );
		file2.close();
		file.clear();
		file.open( filename, ios::out | ios::in | ios::binary );
		if( !file.is_open())
			return 1;
	}
	const int size = 6;
	scan_info data[size]={};
	for( int i=0; i<size; i++ )
	{
		cout << "\nВведите наименование модели -> ";
		cin >> data[i].model;
		cout << "Введите цену -> ";
		cin >> data[i].price;
		cout << "Введите горизонтальный размер области сканирования -> ";
		cin >> data[i].x_size;
		cout << "Введите вертикальный размер области сканирования -> ";
		cin >> data[i].y_size;
		cout << "Введите оптическое разрешение -> ";
		cin >> data[i].optr;
		cout << "Введите число градаций серого -> ";
		cin >> data[i].grey;
	}
	for( int i=0; i<size; i++ )
		safe( file, data[i] );
	cout << "\n\nEnter load structur namber -> ";
	size_t pos;
	cin >> pos;
	scan_info dat;
	if( load( file, dat, pos )!=0 )
		cout << "Error write !\n";
	else
	{
		cout << "\nНаименование модели -> " << dat.model << "\n";
		cout << "Цена -> "<< dat.price << "\n";
		cout << "Горизонтальный размер области сканирования -> "<< dat.x_size
<< "\n";
		cout << "Вертикальный размер области сканирования -> "<< dat.y_size <<
"\n";
		cout << "Оптическое разрешение -> "<< dat.optr << "\n";
		cout << "Число градаций серого -> "<< dat.grey << "\n";
	}
	file.close();
	cin.sync();
	cin.clear();
	std::cin.get();
```

```
        return 0;
}
void safe( fstream & out, const scan_info & data )
{
        out.seekg(0);
        wchar_t size;
        out.read( reinterpret_cast<char*>( &size ), 2 );
        out.seekp(0);
        size++;
        out.write( reinterpret_cast<char*>(&size), 2 );
        out.seekp( 0, ios_base::end );
        out.write((char*) &data, sizeof(scan_info));
        out.flush();
}

int load( fstream & in, scan_info & data, size_t pos )
{
        in.seekg(0);
        in.clear();
        wchar_t size;
        in.read( reinterpret_cast<char*>( &size ), 2 );
        if( !in || int(size) < pos )
                return -1;
        in.seekg(2+ pos*sizeof(scan_info), ios::beg );
        in.read((char*)&data, sizeof(scan_info));
        return 0;
}
```

## Варианты заданий

Вариант 1

Для хранения данных о цветных планшетных сканерах описать структуру вида:

struct scan_info

{

char model[25]:  // наименование подели

int price;  // цена

double x_size;  // горизонтальный размер области сканирования

double y_size;  // вертикальный размер области сканирования

int optr;  // оптическое разрешение

int grey:  // число градаций серого

};

Написать функцию, которая записывает в бинарный файл данные о сканере из приведенной структуры. Структура файла: в первых двух байтах размещается

значение типа int, определяющее количество сделанных в файл записей; далее без пропусков размещаются записи о сканерах.

Написать функцию, которая извлекает из этого файла данные о сканере в структуру типа scarMnfo. Обязательный параметр — номер требуемой записи. Функция должна возвращать нулевое значение, если чтение прошло успешно, и -1 в противном случае.

Привести пример программы, создающей файл с данными о сканерах (данные вводятся с клавиатуры) — 6-8 записей и выводящей на дисплей данные о запрошенной записи.

Все необходимые данные для функций должны передаваться им в качестве параметров. Использование глобальных переменных в функциях не допускается.

Вариант 2

Для хранения данных о планшетных сканерах описать структуру вида:

```
struct scan_info
{
char model[25]:    // наименование подели
int price;   // цена
double x_size;      // горизонтальный размер области сканирования
double y_size;      // вертикальный размер области сканирования
int optr;     // оптическое разрешение
int grey:     // число градаций серого
};
```

Написать функцию, которая записывает в бинарный файл данные о сканере из приведенной структуры. Структура файла: в первых двух байтах размещается значение типа int, определяющее количество сделанных в файл записей; далее без пропусков размещаются записи о сканерах.

Написать функцию, которая сортирует записи в описанном выше бинарном файле по одной из следующих характеристик: цена либо число градаций серого. Обязательный параметр — признак, задающий критерий сортировки.

Привести пример программы, создающей файл с данными о сканерах (данные вводятся с клавиатуры) из не менее восьми записей и осуществляющий его сортировку.

Все необходимые данные для функций должны передаваться им в качестве параметров. Использование глобальных переменных в функциях не допускается.

Вариант 3

Для храпения данных о планшетных сканерах описать структуру вида:

```
struct scan_info
{
char model[25]:    // наименование подели
int price;   // цена
double x_size;      // горизонтальный размер области сканирования
double y_size;      // вертикальный размер области сканирования
int optr;     // оптическое разрешение
int grey:     // число градаций серого
};
```

Написать функцию, которая записывает в бинарный файл данные о сканере из приведенной структуры. Структура файла: в первых четырех байтах размещается значение типа long, определяющее количество сделанных в файл записей; далее без пропусков размещаются записи о сканерах.

Написать функцию, которая сортирует записи в описанном выше бинарном файле по наименованию модели сканера.

Привести пример программы, создающей файл с данными о сканерах (данные вводятся с клавиатуры) из не менее восьми записей и осуществляющий его сортировку.

Все необходимые данные для функций должны передаваться им в качестве параметров. Использование глобальных переменных в функциях не допускается.

Вариант 4

Для храпения данных о планшетных сканерах описать структуру вида:

```
struct scan_info
{
char model[25]:    // наименование подели
int price;   // цена
double x_size;      // горизонтальный размер области сканирования
double y_size;      // вертикальный размер области сканирования
int optr;     // оптическое разрешение
int grey:     // число градаций серого
};
```

Написать функцию, которая динамически выделяет память под массив структур (не меньше шести элементов), заполняет его данными в режиме диалога и записывает массив в бинарный файл. Структура файла: в первых двух байтах размещается значение типа int, определяющее количество сделанных в файл записей; далее без пропусков размещаются записи о сканерах.

Написать функцию, которая извлекает данные о сканере из описанного выше бинарного файла в структуру типа scan_info. Обязательный параметр — номер требуемой записи. Функция должна возвращать пулевое значение, если чтение прошло успешно, и -1 в противном случае.

Привести пример программы, создающей файл с данными о сканерах (данные вводятся с клавиатуры) из не менее восьми записей и осуществляющий вывод па дисплей данных о требуемой записи.

Все необходимые данные для функций должны передаваться им в качестве параметров. Использование глобальных переменных в функциях не допускается.

Вариант 5

Для хранения данных о планшетных сканерах описать структуру вида:

```
struct scan_info
{
char model[25]:    // наименование подели
int price;   // цена
double x_size;      // горизонтальный размер области сканирования
double y_size;      // вертикальный размер области сканирования
int optr;     // оптическое разрешение
int grey:     // число градаций серого
};
```

Написать функцию, которая записывает данные о сканере из приведенной структуры в требуемую позицию в бинарном файле. Структура файла: в первых двух байтах размещается значение типа int, определяющее количество сделанных в файл записей; далее без пропусков размещаются записи о сканерах. Запись может осуществляться в любую позицию, причем если между вводимой записью и последней (или началом файла) имеются пропуски, они заполняются пулями.

Написать функцию, которая «уплотняет» описанный выше бинарный файл путем удаления из него записей, содержащих все нули.

Привести пример программы, создающей файл с данными о сканерах (данные вводятся с клавиатуры) из не менее шести записей и осуществляющий его уплотнение.

Все необходимые данные для функций должны передаваться им в качестве параметров. Использование глобальных переменных в функциях не допускается.

Вариант 6

Для хранения данных о планшетных сканерах описать структуру вида:

```
struct scan_info
{
```

```
char model[25];    // наименование подели
int price;    // цена
double x_size;      // горизонтальный размер области сканирования
double y_size;      // вертикальный размер области сканирования
int optr;     // оптическое разрешение
int grey:     // число градаций серого
};
```

Написать функцию, которая динамически выделяет память под массив структур (не меньше шести элементов), заполняет его данными в режиме диалога и записывает массив в бинарный файл. Структура файла: в первых двух байтах размещается значение типа int, определяющее количество сделанных в файл записей; далее без пропусков размещаются записи о сканерах.

Написать функцию, которая запрашивает данные о сканере в режиме диалога и замещает записи в бинарном файле по заданному номеру. Обязательный параметр — номер замещаемой записи. Функция должна возвращать пулевое значение, если запись прошла успешно, и -1 в противном случае.

Привести пример программы, создающей файл с данными о сканерах (данные вводятся с клавиатуры) из не менее восьми записей и осуществляющий вставку новых данных о сканере.

Все необходимые данные для функций должны передаваться им в качестве параметров. Использование глобальных переменных в функциях не допускается.

## Вариант 7

Для хранения данных о планшетных сканерах описать структуру вида:

```
struct scan_info
{
char model[25]:    // наименование подели
int price;    // цена
double x_size;      // горизонтальный размер области сканирования
```

```
double y_size;      // вертикальный размер области сканирования
int optr;     // оптическое разрешение
int grey:     // число градаций серого
};
```

Написать функцию, которая записывает в бинарный файл данные о сканере из приведенной структуры. Структура файла: в первых двух байтах размещается значение типа int, определяющее количество сделанных в файл записей; далее без пропусков размешаются записи о сканерах.

Написать функцию, которая вводит данные о сканере с клавиатуры в структуру типа scaninfo, и если данные об этом сканере отсутствуют в файле, помещает содержимое структуры в конец файла; в противном случае выдает соответствующее сообщение.

Привести пример программы, создающей файл с данными о сканерах (данные вводятся из текстового файла) — 6-8 записей и дополняющей файл записями о 2-3 сканерах, вводимых с клавиатуры.

Все необходимые данные для функций должны передаваться им в качестве параметров. Использование глобальных переменных в функциях не допускается.

## Вариант 8

Для хранения данных о планшетных сканерах описать структуру вида:

```
struct scan_info
{
char model[25]:    // наименование подели
int price;   // цена
double x_size;      // горизонтальный размер области сканирования
double y_size;      // вертикальный размер области сканирования
int optr;     // оптическое разрешение
int grey:     // число градаций серого
};
```

Написать функцию, которая записывает в бинарный файл данные о сканере из приведенной структуры. Структура файла: в первых двух байтах размещается значение типа int, определяющее количество сделанных в файл записей; далее без пропусков размещаются записи о сканерах.

Написать функцию, которая вводит данные о сканере с клавиатуры в структуру типа scan_info и помещает ее содержимое па место первой записи в файле. Файл должен существовать. При этом, запись ранее занимавшая первую позицию, помещается па вторую, вторая запись на третью, и т. д.

Привести пример программы, создающей файл с данными о сканерах (данные вводятся из текстового файла) — 6-8 записей и дополняющей этот файл 1-2 новыми записями, вводимыми с клавиатуры.

Все необходимые данные для функций должны передаваться им в качестве параметров. Использование глобальных переменных в функциях не допускается.

## Вариант 9

Для хранения данных о планшетных сканерах описать структуру вида:

```
struct scan_info
{
char model[25]:    // наименование подели
int price;   // цена
double x_size;      // горизонтальный размер области сканирования
double y_size;      // вертикальный размер области сканирования
int optr;     // оптическое разрешение
int grey:     // число градаций серого
};
```

Написать функцию, которая запрашивает количество сканеров, информация о которых будет вводиться, динамически выделяет память под массив структур соответствующего размера и заполняет его данными в режиме диалога (с клавиатуры). При этом имя сканера может содержать пробелы.

Написать функцию, которая записывает данный массив в создаваемый бинарный файл. Если цепа сканера меньше 200, то данные об этом сканере в файл не записываются. Информация об остальных сканерах помещается в бинарный файл, причем сначала пишутся данные о всех сканерах, имя которых начинается с заглавной буквы, а затем — с прописной.

Структура файла; в первых четырех байтах размещается значение типа 1 ong, определяющее количество сделанных в файл записей; далее без пропусков размещаются записи о сканерах.

Привести пример программы, создающей файл с данными о сканерах и осуществляющий вывод на дисплей данных о требуемой записи (либо всех, либо по номеру).

Все необходимые данные для функций должны передаваться им в качестве параметров. Использование глобальных переменных в функциях не допускается.

Вариант 10

Для хранения данных о ноутбуках описать структуру вида (при необходимости дополнив ее):

```
struct NOTEBOOK*
{
        char model[21]: // наименование s
        struct size
        {       // габаритные размеры
        float x;
        float y;
        float z;
        };
        float w;                //вес
         int price;             //цена
}
```

Написать функцию, которая читает данные о ноутбуках из файла note.txt в структуру приведенного вида. Написать функцию, которая записывает содержимое структуры в конец бинарного файла. Структура бинарного файла: первые два байта (целое) — число записей в файле; далее записи в формате ■ структуры NOTEBOOK.

Написать программу, в которой на основе разработанных функций осуществляется чтение данных только для тех ноутбуков, частота процессора которых больше 120 МГц, и запись в бинарный файл по убыванию цены.

Вариант 11

Для хранения данных о ноутбуках описать структуру вида:

```
struct NOTEBOOK*
{
        char model[21]: // наименование s
        struct size
        {       // габаритные размеры
        float x;
        float y;
        float z;
        };
        float w;                //вес
         int price;             //цена
}
```

Написать функцию, которая читает данные о ноутбуках из файла note.txt в структуру приведенного вида. Написать функцию, которая записывает содержимое структуры в конец бинарного файла. Структура бинарного файла: первые два байта (целое) — число записей в файле; далее записи в формате структуры NOTEBOOK.

Написать программу, в которой па основе разработанных функций осуществляется чтение данных только для тех ноутбуков, объем HDD которых

меньше 1 Гбайт, и запись считанных данных в бинарный файл в алфавитном порядке по наименованию.

Вариант 12

Для хранения данных о ноутбуках описать структуру вида:

```
struct NOTEBOOK*
{
        char model[21]: // наименование s
        struct size
        {       // габаритные размеры
        float x;
        float y;
        float z;
        };
        float w;                //вес
         int price;             //цена
}
```

Написать функцию, которая читает данные о ноутбуках из файла note.txt в структуру приведенного вида. Написать функцию, которая записывает содержимое структуры в конец бинарного файла. Структура бинарного файла: первые два байта (целое) — число записей в файле; далее записи в формате структуры NOTEBOOK.

Написать программу, в которой на основе разработанных функций осуществляется запись в двоичный файл данных только о тех ноутбуках, целое количество которых в одном кубическом метре не превышает 285 штук.

Вариант 13

Для хранения данных о ноутбуках описать структуру вида:

```
struct NOTEBOOK*
{
        char model[21]: // наименование s
        struct size
        {       // габаритные размеры
        float x;
        float y;
        float z;
        };
        float w;                //вес
         int price;             //цена
}
```

Написать функцию, которая читает данные о ноутбуках из файла note.txt в структуру приведенного вида. Написать функцию, которая записывает содержимое структуры в конец бинарного файла. Структура бинарного файла: первые два байта (целое) — число записей в файле; далее записи в формате структуры NOTEBOOK.

Написать программу, в которой на основе разработанных функций осуществляется запись в двоичный файл данных только о тех ноутбуках, максимальный объем ОЗУ которых не менее 40 Мбайт, отсортированных по объему.

## Вариант 14

Для храпения данных о ноутбуках описать структуру вида:

```
struct NOTEBOOK*
{
        char model[21]: // наименование s
        struct size
        {       // габаритные размеры
        float x;
```

```
        float y;
        float z;
        };
        float w;            //вес
         int price;         //цена
}
```

Написать функцию, которая читает данные о ноутбуках из файла note.txt в структуру приведенного вида. Написать функцию, которая записывает содержимое структуры в конец бинарного файла. Структура бинарного файла: первые два байта — целое число записей в файле; далее записи в формате структуры NOTEBOOK.

Написать программу, в которой на основе разработанных функций осуществляется запись в двоичный файл данных только о тех ноутбуках, диагональ дисплея которых больше одиннадцати дюймов.

Вариант 15

Для храпения данных о ноутбуках описать структуру вида (при необходимости дополнив ее):

```
struct NOTEBOOK
{
struct disp_res
{       // разрешающая способность дисплея
int x;  // по горизонтали
int y;  // по вертикали
}
int f;
float d; int price;
char model[21];
//      частота регенерации
//      размер диагонали дисплея
```

```
//      цена
//      наименование
}
```

Написать функцию, которая читает данные о ноутбуках из файла note.txt в структуру приведенного вида. Написать функцию, которая записывает содержимое структуры в конец бинарного файла. Структура бинарного файла: первые два байта — целое число записей в файле; далее записи в формате структуры NOTEBOOK.

Написать программу, в которой па основе разработанных функций осуществляется запись в двоичный файл данных только о тех ноутбуках, вес которых менее 7 кг, отсортированных в порядке возрастания цепы.

## Вариант 16

Для храпения данных о ноутбуках описать структуру вида:

```
struct NOTEBOOK
{
struct disp_res
{       // разрешающая способность дисплея
int x;  // по горизонтали
int y;  // по вертикали
}
int f;
float d; int price;
char model[21];
//      частота регенерации
//      размер диагонали дисплея
//      цена
//      наименование
}
```

Написать функцию, которая читает данные о ноутбуках из файла note.txt в структуру приведенного вида. Написать функцию, которая записывает содержимое структуры в конец бинарного файла. Структура бинарного файла: первые два байта — целое число записей в файле; далее записи в формате структуры NOTEBOOK.

Написать программу, в которой на основе разработанных функций осуществляется запись в двоичный файл данных только о тех ноутбуках, объем видеопамяти которых 2 Мбайт, отсортированных в порядке уменьшения тактовой частоты процессора.

Вариант 17

Для храпения данных о ноутбуках описать структуру вида:

```
struct NOTEBOOK
{
struct disp_res
{       // разрешающая способность дисплея
int x;  // по горизонтали
int y;  // по вертикали
}
int f;
float d; int price;
char model[21];
//      частота регенерации
//      размер диагонали дисплея
//      цена
//      наименование
}
```

Написать функцию, которая читает данные о ноутбуках из файла note.txt в структуру приведенного вида. Написать функцию, которая записывает содержимое

структуры в конец бинарного файла. Структура бинарного файла: первые два байта — целое число записей в файле; далее записи в формате структуры NOTEBOOK.

Написать программу, в которой па основе разработанных функций осуществляется запись в двоичный файл данных только о тех ноутбуках, объем HDD которых больше 1 Гбайт, отсортированных в порядке возрастания размера диагонали дисплея.

Вариант 18

Для хранения данных о ноутбуках описать структуру вида:

```
struct NOTEBOOK
{
struct disp_res
{       // разрешающая способность дисплея
int x;  // по горизонтали
int y;  // по вертикали
}
int f;
float d; int price;
char model[21];
//      частота регенерации
//      размер диагонали дисплея
//      цена
//      наименование
}
```

Написать функцию, которая читает данные о ноутбуках из файла note.txt в структуру приведенного вида. Написать функцию, которая записывает содержимое структуры в конец бинарного файла. Структура бинарного файла: первые два байта — целое число записей в файле; далее записи в формате структуры NOTEBOOK.

Написать программу, в которой па основе разработанных функций осуществляется запись в двоичный файл данных только о тех ноутбуках, тактовая частота процессора которых больше 120МГц, отсортированных в порядке уменьшения веса.

Вариант 19

Для храпения данных о ноутбуках описать структуру вида (при необходимости дополнив ее):

```
struct NOTEBOOK
{
struct disp_res
{ // разрешающая способность дисплея
int x;  // по горизонтали
int y;  // по оертикали
};
int f;      // частота регенерации
float d;     // размер диагонали дисплея
float held;    // объем диска
char model[21]; // наименование
}
```

Написать функцию, которая читает данные о ноутбуках из файла note.txt в структуру приведенного вида. Написать функцию, которая записывает содержимое структуры в конец бинарного файла. Структура бинарного файла: первые два байта — целое число записей в файле; далее записи в формате структуры NOTEBOOK. Написать программу, в которой па основе разработанных функций осуществляется запись в двоичный файл данных только о тех ноутбуках, тактовая частота процессора которых больше 120МГц, отсортированные в порядке возрастания цены.

Вариант 20

Для храпения данных о ноутбуках описать структуру вида:

```
struct NOTEBOOK
{
struct disp_res
{       // разрешающая способность дисплея
int x;  // по горизонтали
int y;  // по вертикали
}
int f;
float d; int price;
char model[21];
//      частота регенерации
//      размер диагонали дисплея
//      цена
//      наименование
}
```

Написать функцию, которая читает данные о ноутбуках из файла note.txt в структуру приведенного вида. Написать функцию, которая записывает содержимое структуры в конец бинарного файла. Структура бинарного файла: первые два байта — целое число записей в файле; далее записи в формате структуры NOTEBOOK. Написать программу, в которой на основе разработанных функций осуществляется запись в двоичный файл данных только о тех ноутбуках, цепа которых больше $3500, отсортированные в порядке возрастания тактовой частоты процессора. Пример файла note.txt:

```
Acer Note Light       2699 5.6 02.0x11.8x08.3  100 40 10.4 1 1024x0768          60
      0.774
ASW ND5123T           3489 7.2 02.3x11.8x10.1  133 32 12.1 2 1024x0768          70
      1.300
```

ARMNote TS80CD 3699 7.2 02.0x11.5x08.8 133 64 11.3 1 1024x0768 75 1.300

AST Ascentia P50 4499 7.5 02.3x11.3x09.0 133 40 11.3 1 0800x0600 70 0.774

BSI NP8657D 2605 8.0 02.3x11.8x09.3 133 40 11.3 1 1024x0768 600.810

BSI NP5265A 3765 8.2 02.5x12.0x09.0 150 32 12.1 2 1024x0768 70 1.300

Dell Xpi P100SD 3459 6.0 02.3x11.0x08.8 100 40 10.3 1 1024x0768 60 0.773

Digital HiNote 4799 4.0 01.3x11.0x08.8 120 40 10.4 1 0800x0600 56 1.000

Gateway Solo S5 4499 5.6 02.0x11.9x08.8 133 40 11.3 2 1024x0768 60 0.686

Hertz Z-Optima NB 3995 8.0 02.3x11.9x09.0 150 40 11.2 2 1024x0768 75 1.000

HP OmniBook 5500 6120 7.1 02.0x11.5x09.0 133 64 11.4 1 1024x0768 75 1.300

IBM ThinkPad 560 3749 4.1 01.3x11.8x08.8 120 40 12.1 2 1024x0768 85 0.774

NEC Versa 4080H 4780 6.6 02.3x11.8x09.5 120 48 10.4 1 0800x0600 70 0.776

Polywell Poly 500 3300 7.9 02.3x11.9x09.0 120 40 10.4 1 1024x0768 72 1.000

Samsung SENS 810 3667 8.7 02.3x11.5x09.5 100 32 11.4 2 1024x0768 75 0.773

Twinhead Slimnote 2965 7.4 02.0x11.5x08.0 075 64 10.4 1 1024x0768 70 0.772

В файле note.txt находится текстовая информация о ноутбуках. Каждая строка содержит данные об одной модели. Данные в строке размещаются в следующих полях:

1 : 20 — наименование модели;

21 : 24 — цена в долларах (целое число);

26 : 28 — масса ноутбука в кг (число с десятичной точкой из четырех символов);

30 : 43 — габаритные размеры ноутбука в дюймах (ВЫСОТАхДЛИНАхШИРИНА — три числа с десятичной точкой (4 символа, включая точку, разделенные 'x');

44 : 47 — частота процессора в МГц (целое число из трех символов); 49 :50 — максимальный объем ОЗУ в мегабайтах (целое число из двух символов); 52 : 55 — размер диагонали дисплея в дюймах (число с десятичной точкой из четырех символов, включая точку);

57 — размер видеопамяти в мегабайтах — целое число из одного символа; 59 : 67 — разрешающая способность дисплея в пикселах (два целых числа, разделенные 'x');

69 : 70 — частота регенерации дисплея в Гц (целое число из двух символов); 72 :76 — объем HDD в гигабайтах (число с десятичной точкой из пяти символов). Все неописанные позиции заполнены пробелами.

- 

## Лабораторная работа № 6. Создание и использование классов

Цель работы – отработка умений и навыков создания абстрактного класса на языке C++.

### Методические указания

Для выполнения лабораторной работы необходимо изучить следующие разделы курса лекций:

- 3.1 Классы.

*Класс* является абстрактным типом данных, определяемым пользователем, и представляет собой модель реального объекта в виде данных и функций для работы с ними.

Данные класса называются *полями* (но аналогии с полями структуры), а функции класса — *методами*. Поля и методы называются *элементами класса*.

```
class <имя>
{
[ private:]
    <описание скрытых элементов>
public:
    <описание доступных элементов>
};
```

### Пример выполнения лабораторной работы

*Задание:*

Описать класс, реализующий стек. Написать программу, использующую этот класс для моделирования Т-образного сортировочного узла на железной дороге. Программа должна разделять на два направления состав, состоящий из вагонов двух типов (на каждое направление формируется состав из вагонов одного типа). Предусмотреть возможность формирования состава из файла и с клавиатуры.

*Листинг программы:*

```
#include <iostream>
#include <stdlib.h>
#include <time.h>
#include <stdio.h>
#include <conio.h>
#include <string.h>
using namespace std;

class Stek
{
	int m[100],
		chet[100],
		nechet[100],
		n,
		kch,
		kn;
public:
	void iz_fail();
	void s_klav();
	void prosm();
	void razdelenie();
	void prosm_chet();
	void prosm_nechet();
};

void main()
{
	setlocale(0, "rus");
	Stek s;
	int vibor;
	vibor = rand();
	system("cls");

	while (vibor!=7)
	{
		cout<<"\n"<<endl;
		cout<<"Выберите нужный пункт меню:";cout<<endl;
		cout<<"\n1. Заполнить стек из файла;";cout<<endl;
		cout<<"2. Заполнить с клавиатуры;";cout<<endl;
		cout<<"3. Просмотреть содержимое стека;";cout<<endl;
		cout<<"4. Разделение содержимого стека;";cout<<endl;
		cout<<"5. Просмотреть массив из четных данных;";cout<<endl;
		cout<<"6. Просмотреть массив из нечетных данных;";cout<<endl;
		cout<<"7. Закончить выбор пунктов меню.";cout<<endl;
		cout<<"\n";
		cin>>vibor;
		switch (vibor)
		{
		case 1:{s.iz_fail();break;}
		case 2:{s.s_klav();break;}
		case 3:{s.prosm();break;}
		case 4:{s.razdelenie();break;}
		case 5:{s.prosm_chet();break;}
		case 6:{s.prosm_nechet();break;}
		case 7:{break;}
		default:{cout<<"Такого пункта меню нет!"<<endl;}
		}
	}
}

void Stek::iz_fail()
{
	FILE *f;
```

```
        int a,i;
        f=fopen("s", "wb");
        cout<<"Заполнение файла данными -";
        cout<<endl;
        cout << "Введите количество элементов: ";
        cin >> n;
        for (i=1; i<=n; i++)
        {
                a = (rand() % 50)-25;
                fwrite(&a, sizeof(int), 1, f);
        }
        fclose(f);
        f=fopen("s", "rb");
        i=0;
        while (fread(&a,sizeof(int),1,f))
        {
                m[i]=a;
                i++;
        }
        fclose(f);
        cout<<endl<<"Файл заполнен!!!"<<endl;
};

void Stek::s_klav()
{
        int i,a;
        cout << "Введите количество элементов: ";
        cin >> n;
        for (i=1; i<=n; i++)
        { cout << "Значение элемента: ";
        cin>>a;
        m[i-1]=a;
        }
}

void Stek::prosm()
{
        int i;
        if (n==0) cout<<"Элементов в стеке нет!";
        else
        {
                cout<<"Содержимое стека на данном этапе:";cout<<endl;
                for (i=0; i<n; i++) { cout<<m[i]<<" "; }
                cout<<"Просмотр выполнен!!!"<<endl;
        }
}

void Stek::razdelenie()
{
        int i;
        if (n==0) cout<<" Элементов в стеке нет!";
        else
        {
                i=n-1;
                kch=0;
                kn=0;
                while (i>=0)
                {
                        if (m[i]%2==0) {chet[kch]=m[i];kch++;n--;}
                        else {nechet[kn]=m[i];kn++;n--;}
                        i--;
                }
                cout<<"Разделение выполнено!!!"<<endl;
```

```
        }
}

void Stek::prosm_chet()
{
        int i;
        if (kch==0) cout<<"Элементов в массиве нет!";
        else
        {
                cout<<"Содержимое массива четных чисел:"<<endl;
                for  (i=0; i<kch; i++)
                {
                        cout<<chet[i]<<" ";
                }
        }
}

void Stek::prosm_nechet()
{
        int i;
        if (kn==0) cout<<" Элементов в массиве нет!";
        else
        {
                cout<<" Содержимое массива нечетных чисел"<<endl;
                for  (i=0; i<kn; i++)
                {
                        cout<<nechet[i]<<" ";
                }
        }
}
```

## Варианты заданий

### Вариант 1

Описать класс, реализующий стек. Написать программу, использующую этот класс для моделирования Т-образного сортировочного узла на железной дороге. Программа должна разделять на два направления состав, состоящий из вагонов двух типов (на каждое направление формируется состав из вагонов одного типа). Предусмотреть возможность формирования состава из файла и с клавиатуры.

### Вариант 2

Описать класс, реализующий бинарное дерево, обладающее возможностью добавления новых элементов, удаления существующих, поиска элемента по ключу, а также последовательного доступа ко всем элементам.

Написать программу, использующую этот класс для представления англо-русского словаря. Программа должна содержать меню, позволяющее осуществить проверку всех методов класса. Предусмотреть возможность формирования словаря из файла и с клавиатуры.

Вариант 3

Построить систему классов для описания плоских геометрических фигур: круг, квадрат, прямоугольник. Предусмотреть методы для создания объектов, перемещения на плоскости, изменения размеров и вращения на заданный угол.

Написать программу, демонстрирующую работу с этими классами. Программа должна содержать меню, позволяющее осуществить проверку всех методов классов.

Вариант 4

Построить описание класса, содержащего информацию о почтовом адресе организации. Предусмотреть возможность раздельного изменения составных частей адреса, создания и уничтожения объектов этого класса.

Написать программу, демонстрирующую работу с этим классом. Программа должна содержать меню, позволяющее осуществить проверку всех методов класса.

Вариант 5

Составить описание класса для представления комплексных чисел. Обеспечить выполнение операций сложения, вычитания и умножения комплексных чисел. Написать программу, демонстрирующую работу с этим классом. Программа должна содержать меню, позволяющее осуществить проверку всех методов класса.

Вариант 6

Составить описание класса для объектов-векторов, задаваемых координатами концов в трехмерном пространстве. Обеспечить операции сложения и вычитания векторов с получением нового вектора (суммы или разности), вычисления скалярного произведения двух векторов, длины вектора, косинуса угла между векторами.

Написать программу, демонстрирующую работу с этим классом. Программа должна содержать меню, позволяющее осуществить проверку всех методов класса.

Вариант 7

Составить описание класса прямоугольников со сторонами, параллельными осям координат. Предусмотреть возможность перемещения прямоугольников на плоскости, изменения размеров, построения наименьшего прямоугольника, содержащего два заданных прямоугольника, и прямоугольника, являющегося общей частью (пересечением) двух прямоугольников.

Написать программу, демонстрирующую работу с этим классом. Программа должна содержать меню, позволяющее осуществить проверку всех методов класса.

Вариант 8

Составить описание класса для определения одномерных массивов целых чисел (векторов). Предусмотреть возможность обращения к отдельному элементу массива с контролем выхода за пределы массива, возможность задания произвольных границ индексов при создании объекта и выполнения операций поэлементного сложения и вычитания массивов с одинаковыми границами индексов, умножения и деления всех элементов массива на скаляр, вывода на экран элемента массива по заданному индексу и всего массива.

Написать программу, демонстрирующую работу с этим классом. Программа должна содержать меню, позволяющее осуществить проверку всех методов класса.

Вариант 9

Составить описание класса для определения одномерных массивов строк фиксированной длины. Предусмотреть контроль выхода за пределы массива, возможность обращения к отдельным строкам массива по индексам, выполнения операций поэлементного сцепления двух массивов с образованием нового массива, слияния двух массивов с исключением повторяющихся элементов, а также вывод на экран элемента массива по заданному индексу и всего массива.

Написать программу, демонстрирующую работу с этим классом. Программа должна содержать меню, позволяющее осуществить проверку всех методов класса.

Вариант 10

Составить описание класса многочленов от одной переменной, задаваемых степенью многочлена и массивом коэффициентов. Предусмотреть методы для вычисления значения многочлена для заданного аргумента, операции сложения, вычитания и умножения многочленов с получением нового объекта-многочлена, вывод на экран описания многочлена.

Написать программу, демонстрирующую работу с этим классом. Программа должна содержать меню, позволяющее осуществить проверку всех методов класса.

Вариант 11

Составить описание класса одномерных массивов строк, каждая строка задается длиной и указателем на выделенную для нее память. Предусмотреть контроль выхода за пределы массивов, возможность обращения к отдельным строкам массива по индексам, выполнения операций поэлементного сцепления двух массивов с образованием нового массива, слияния двух массивов с исключением повторяющихся элементов, а также вывод на экран элемента массива и всего массива.

Написать программу, демонстрирующую работу с этим классом. Программа должна содержать меню, позволяющее осуществить проверку всех методов класса.

Вариант 12

Составить описание класса, обеспечивающего представление матрицы произвольного размера с возможностью изменения числа строк и столбцов, вывода на экран подматрицы любого размера и всей матрицы.

Написать программу, демонстрирующую работу с этим классом. Программа должна содержать меню, позволяющее осуществить проверку всех методов класса.

Вариант 13

Написать класс для эффективной работы со строками, позволяющий форматировать и сравнивать строки, хранить в строках числовые значения и извлекать их. Для этого необходимо реализовать:

- перегруженные операторы присваивания и конкатенации;
- операции сравнения и приведения типов;
- преобразование в число любого типа;
- форматный вывод строки.

Написать программу, демонстрирующую работу с этим классом. Программа должна содержать меню, позволяющее осуществить проверку всех методов класса.

Вариант 14

Описать класс «домашняя библиотека». Предусмотреть возможность работы с произвольным числом книг, поиска книги по какому-либо признаку

(например, по автору или по году издания), добавления книг в библиотеку, удаления книг из нее, сортировки книг по разным полям.

Написать программу, демонстрирующую работу с этим классом. Программа должна содержать меню, позволяющее осуществить проверку всех методов класса.

Вариант 15

Описать класс «записная книжка». Предусмотреть возможность работы с произвольным числом записей, поиска записи по какому-либо признаку (например, о фамилии, дате рождения или номеру телефона), добавления и удаления записей, сортировки по разным нолям.

Написать программу, демонстрирующую работу с этим классом. Программа должна содержать меню, позволяющее осуществить проверку всех методов класса.

Вариант 16

Описать класс «студенческая группа». Предусмотреть возможность работы с непременным числом студентов, поиска студента по какому-либо признаку (например, по фамилии, дате рождения или номеру телефона), добавления и удаления записей, сортировки по разным полям.

Написать программу, демонстрирующую работу с этим классом. Программа должна содержать меню, позволяющее осуществить проверку всех методов класса.

Вариант 17

Описать класс, реализующий тип данных «вещественная матрица» и работу с ними. Класс должен реализовывать следующие операции над матрицами:

- сложение, вычитание, умножение, деление (+, -, *, /) (умножение и деление, как на другую матрицу, так и на число);
- комбинированные операции присваивания (+=, -=, *=, /=);

- операции сравнения на равенство/неравенство;
- операции вычисления обратной и транспонированной матрицы, операцию возведения в степень; методы вычисления детерминанта и нормы; методы, реализующие проверку типа Матрицы (квадратная, диагональная, нулевая, единичная, симметрическая, верхняя треугольная, нижняя треугольная);
- операции ввода/вывода в стандартные потоки.

Написать программу, демонстрирующую работу с этим классом. Программа должна содержать меню, позволяющее осуществить проверку всех методов класса.

Вариант 18

Описать класс «множество», позволяющий выполнять основные операции — добавление и удаление элемента, пересечение, объединение и разность множеств.

Написать программу, демонстрирующую работу с этим классом. Программа должна содержать меню, позволяющее осуществить проверку всех методов класса.

Вариант 19

Описать класс, реализующий стек. Написать программу, использующую этот класс для отыскания прохода по лабиринту.

Лабиринт представляется в виде матрицы, состоящей из квадратов. Каждый квадрат либо открыт, либо закрыт. Вход в закрытый квадрат запрещен. Если квадрат открыт, то вход в него возможен со стороны, но не с угла. Каждый квадрат определяется его координатами в матрице. После отыскания прохода программа печатает найденный путь в виде координат квадратов.

Вариант 20

Описать класс «предметный указатель». Каждая компонента указателя содержит слово и номера страниц, на которых это слово встречается.

Количество номеров страниц, относящихся к одному слову, от одного до десяти. Предусмотреть возможность формирования указателя с клавиатуры и из файла, вывода указателя, вывода номеров страниц для заданного слова, удаления элемента из указателя.

Написать программу, демонстрирующую работу с этим классом. Программа должна содержать меню, позволяющее осуществить проверку всех методов класса.

## Лабораторная работа № 7. Наследование

Цель работы – отработка умений и навыков работы с механизмом наследования классов.

## Методические указания

Для выполнения лабораторной работы необходимо изучить следующие разделы курса лекций:

- 3.2 Наследование.

Механизм наследования классов позволяет строить иерархии, в которых производные классы получают элементы родительских, или базовых, классов и могут дополнять их или изменять их свойства. При большом количестве никак не связанных классов управлять ими становится невозможным. Наследование позволяет справиться с этой проблемой путем упорядочивания и ранжирования классов, то есть объединения общих для нескольких классов свойств в одном классе и использования его в качестве базового.

Классы, находящиеся ближе к началу иерархии, объединяют в себе наиболее общие черты для всех нижележащих классов. По мере продвижения вниз по иерархии классы приобретают все больше конкретных черт. Множественное наследование позволяет одному классу обладать свойствами двух и более родительских классов.

## Пример выполнения лабораторной работы

*Задание:*

Создать абстрактный класс CVehicle. На его основе реализовать классы CPlane, CCar и CShip. Классы должны иметь возможность задавать и получать координаты, параметры средств передвижения (цена, скорость, год выпуска). Для самолета должна быть определена высота, для самолета и корабля — количество пассажиров. Для корабля — порт приписки.

Написать программу, создающую список объектов этих классов в динамической памяти. Программа должна содержать меню, позволяющее осуществить проверку всех методов классов.

*Листинг программы:*

```
#include <list>
#include <string>
#include <iostream>
#include <algorithm>
using namespace std;

class CVehicle abstract
{
public:
	CVehicle(double x, double y, int price, int year, int speed): x_(x), y_(y),
		price_(price), year_(year), speed_(speed) {};
	double getX() { return x_; };
	double getY() { return y_; };
	int getPrice() { return price_; };
	int getYear() { return year_; };
	int getSpeed() { return speed_; }
	void setX(double x) { x_ = x; };
	void setY(double y) { y_ = y; };
	void setYear(int year) { year_ = year; };
	void setPrice(int price) { price_ = price; };
	void setSpeed(int speed) { speed_ = speed; };
	virtual void printString() = 0;
protected:
	double x_, y_;
	int price_;
	int year_;
	int speed_;
};

class CPlane : public CVehicle {
public:
	CPlane(double x, double y, int price, int year, int speed, int height, int
seats): height_(height),
		seats_(seats), CVehicle(x, y, price, year, speed) {};
	int getHeight() { return height_; }
	int getSeats() { return seats_; }
	void setHeight(int height) { height_ = height; }
	void setSeats(int seats) { seats_ = seats; }
	void printString()
	{
		cout << "Самолёт, параметры:\nx: " << x_ << ", y: " << y_ << ", цена: "
<< price_ << ", год: " << year_
			<< ", скорость:" << speed_ << ", высота: " << height_ << ",
количество мест: " << seats_ << ".\n" << endl;
	}
private:
	int height_;
	int seats_;
};

class CCar : public CVehicle {
public:
	CCar(double x, double y, int price, int year, int speed):
	  CVehicle(x, y, price, year, speed) {};
```

```
        void printString()
        {
              cout << "Автомобиль, параметры:\nx: " << x_ << ", y: " << y_ << ",
цена: " << price_ << ", год: " << year_
                   << ", скорость:" << speed_ << ".\n" << endl;
        }
};

class CShip : public CVehicle {
public:
      CShip(double x, double y, int price, int year, int speed, int seats, string
port): port_(port),
            seats_(seats), CVehicle(x, y, price, year, speed) {};
      string getPort(){ return port_; }
      int getSeats(){ return seats_; }
      void setSeats(int seats) { seats_ = seats; }
      void setPort(string port) { port_ = port; }
      void printString()
      {
            cout << "Корабль, параметры:\nx: " << x_ << ", y: " << y_ << ", цена: "
<< price_ << ", год: " << year_
                 << ", скорость:" << speed_ << ", порт: " << port_ << ",
количество мест: " << seats_ << ".\n" << endl;
      }
private:
      int seats_;
      string port_;
};

template <typename T>
void xor_swap(T& x, T&y) {
      x ^= y ^= x ^= y;
}

int main()
{
      srand(12345);
      setlocale(LC_ALL, "Russian");
      list<CVehicle*> vehicles;
      for (int i = 0; i < 10; ++i) {
            double random = (rand() % 100) / 100.0;
            double x = rand() % 1000;
            double y = rand() % 1000;
            int price = rand() % 1000000;
            int year = rand() % 100 + 1900;
            int speed = rand() % 100;
            if (random <= 0.33)
                  vehicles.push_front(new CCar(x, y, price, year, speed));
            else if (random >= 0.66)
                  vehicles.push_front(new CPlane(x, y, price, year, speed, rand() %
100 , rand() % 100));
            else
            {
                  string port;
                  if (random > 0.5)
                        port = "Каир";
                  else
                        port = "Нальчик";
                  vehicles.push_front(new CShip(x, y, price, year, speed, rand() %
100, port));
            }
      }
      bool cont = true;
```

```
	while (cont)
	{
		cout << "\nМЕНЮ:" << endl << "0 - Выход" << endl << "1 - Вывести список (проверка метода printString())" << endl
		<< "2 - Задать параметры (проверка методов set) " << endl << "3 - Получить параметры \n" ;
		int choice;
		cin >> choice;
		switch (choice)
		{
		case 0:
			cont = false;
			break;
		case 1:
			for (auto it = vehicles.cbegin(); it != vehicles.cend(); ++it)
				(*(*it)).printString();
			break;
		case 2:
			{
				cout << "Для какого из элементов задать параметры?\n";
				int num;
				cin >> num;
				auto it = vehicles.begin();
				int vlength = 0;
				for (auto iter = vehicles.begin(); iter != vehicles.end(); ++iter)
					++vlength;
				if (num > vlength) {
					cout << "Неверное число!" << endl;
					break;
				}
				advance(it, num);

				cout << endl << "   x = ";
				double x;
				cin >> x;
				(*(*it)).setX(x);

				cout <<  "   y = ";
				double y;
				cin >> y;
				(*(*it)).setY(y);

				cout << "   цена = ";
				int price;
				cin >> price;
				(*(*it)).setPrice(price);

				cout << "   скорость = ";
				int speed;
				cin >> speed;
				(*(*it)).setSpeed(speed);

				cout << "   год = ";
				int year;
				cin >> year;
				(*(*it)).setYear(year);
				CPlane* plane = dynamic_cast<CPlane*>(*it);
				if (plane != 0) {
					cout << "   высота = ";
					int height;
					cin >> height;
```

```
                                plane->setHeight(height);


                                cout << "   количество мест = ";
                                int seats;
                                cin >> seats;
                                plane->setHeight(seats);

                        }
                        CShip* ship = dynamic_cast<CShip*>(*it);
                        if (ship != 0) {
                                cout << "   порт = ";
                                string port;
                                cin >> port;
                                ship->setPort(port);


                                cout << "   количество мест = ";
                                int seats;
                                cin >> seats;
                                ship->setSeats(seats);
                        }
                }
                break;
        case 3:
                {
                        cout << "Для какого из элементов получить параметры?\n";
                        size_t num;
                        cin >> num;
                        auto it = vehicles.begin();
                        size_t vlength = 0;
                        for (auto iter = vehicles.begin(); iter != vehicles.end();
++iter)
                                ++vlength;
                        if (num > vlength) {
                                cout << "Неверное число!" << endl;
                                break;
                        }
                        advance(it, num);
                        (*(*it)).printString();
                }
                break;
        }
    };
    for (auto it = vehicles.begin(); it != vehicles.end(); ++it)
        delete (*it);
    return 0;
}
```

## Варианты заданий

### Вариант 1

Создать класс CFile, инкапсулирующий в себе такие функции работы с файлами, как Open, Close, Seek, Read, Write, GetPosition и GetLength. На базе этого класса создать производный класс CMyDataFile — файл, содержащий в

себе данные некоторого определенного типа My Data, а также заголовок, облегчающий доступ к этому файлу.

Написать программу, демонстрирующую работу с этими классами. Программа должна содержать меню, позволяющее осуществить проверку всех методов классов.

Вариант 2

Создать класс CPoint — точка. На его основе создать классы CcoloredPoint и CLine. На основе класса CLine создать класс CColoredLine и класс CPolyLine — многоугольник. Все классы должны иметь методы для установки и получения значений всех координат, а также изменения цвета и получения текущего цвета.

Написать демонстрационную программу, в которой будет использоваться список объектов этих классов в динамической памяти.

Вариант 3

Создать абстрактный класс CVehicle. На его основе реализовать классы CPlane, CCar и CShip. Классы должны иметь возможность задавать и получать координаты, параметры средств передвижения (цена, скорость, год выпуска). Для самолета должна быть определена высота, для самолета и корабля — количество пассажиров. Для корабля — порт приписки.

Написать программу, создающую список объектов этих классов в динамической памяти. Программа должна содержать меню, позволяющее осуществить проверку всех методов классов.

Вариант 4

1. Описать базовый класс «Элемент». Поля:

- имя элемента (указатель на строку символов);
- количество входов элемента;

- количество выходов элемента.

Методы:

- конструктор класса;
- деструктор класса;
- метод, задающий имя элемента.

2. На основе класса «Элемент» описать производный класс «Комбинационный», представляющий собой комбинационный элемент (двоичный вентиль), который может иметь несколько входов и один выход.

Поля:

- указатель, используемый для динамического размещения полей, содержащих значения входов.

Методы:

- конструктор;
- конструктор копирования;
- деструктор;
- метод, задающий значение на входах экземпляра класса;
- метод, позволяющий опрашивать состояние отдельного входа экземпляра класса;
- метод, вычисляющий значение выхода (по варианту задания).

3. На основе класса «Элемент» описать производный класс «Память», представляющих собой триггер. Триггер имеет входы, соответствующие типу триггера (см. ниже вариант задания), и входы установки и сброса. Все триггеры считаются синхронными, сам синхровход в состав триггера не включается.

Поля:

- массив значений входов объекта класса (задается статически), в массиве учитываются все входы (управляющие и информационные);
- состояние на прямом выходе триггера;
- состояние на инверсном выходе триггера.

Методы:

- конструктор (по умолчанию сбрасывает экземпляр класса);
- конструктор копирования;
- деструктор;
- метод, задающий значение на входах экземпляра класса;
- методы, позволяющие опрашивать состояния отдельного входа экземпляра класса;
- метод, вычисляющий состояние экземпляра класса (по варианту задания) в зависимости от текущего состояния и значений на входах;
- метод, переопределяющий операцию == для экземпляров класса.

4. Создать класс «Регистр», используя класс «Память» как включаемый класс.

Поля:

- состояние входа «Сброс» — один для экземпляра класса;
- состояние входа «Установка» - один для экземпляра класса;
- статический массив типа «Память» заданной в варианте размерности;
- статический(е) массив(ы), содержащие значения на соответствующих входах элементов массива типа «Память».

Методы:

- метод, задающий значение на входах экземпляра класса (желательно в качестве параметров передавать методу указатели на массивы значений);
- метод, позволяющий опрашивать состояние отдельного выхода экземпляра класса;
- метод, вычисляющий значение нового состояния экземпляра класса.

Все поля классов «Элемент», «Комбинационный» и «Память» должны быть описаны с ключевым словом private, или protected.

В задании перечислены только обязательные члены и методы класса. Можно задавать дополнительные члены и методы, если они не отменяют обязательные и обеспечивают дополнительные удобства при работе с данными классами, например, описать функции вычисления выхода/состояния как виртуальные.

5. Для проверки функционирования созданных классов написать программу, использующую эти классы. В программе должны быть продемонстрированы все свойства созданных классов.

Конкретный тип комбинационного элемента, тип триггера и разрядность регистра выбираются в соответствии с вариантом задания.

| Вариант | Комбинационный элемент | Число входов | Триггер | Разрядность регистра |
|---|---|---|---|---|
| 1 | И-НЕ | 4 | RS | 8 |
| 2 | ИЛИ | 5 | RST | 10 |
| 3 | МОД2-НЕ | 6 | D | 12 |
| 4 | И | 8 | T | 8 |
| 5 | ИЛИ-НЕ | 8 | V | 9 |
| 6 | И | 4 | RS | 10 |
| 7 | ИЛИ-НЕ | 5 | JK | 11 |
| 8 | МОД2 | 5 | D | 8 |
| 9 | И | 4 | T | 10 |
| 10 | ИЛИ | 3 | JK | 8 |
| 11 | И-НЕ | 3 | RS | 12 |
| 12 | ИЛИ-НЕ | 4 | RST | 4 |
| 13 | МОД2 | 5 | D | 10 |
| 14 | МОД2-НЕ | 6 | T | 10 |
| 15 | ИЛИ-НЕ | 8 | V | 10 |

| | | | | |
|---|---|---|---|---|
| 16 | И | 8 | JK | 6 |
| 17 | И-НЕ | 8 | RS | 10 |
| 18 | ИЛИ | 8 | T | 10 |
| 19 | МОД2 | 6 | JK | 8 |
| 20 | МОД2-НЕ | 5 | V | 10 |

## Лабораторная работа № 8. Функции работы со строками

Цель работы – отработка умений, навыков создания и использования функций работы со строками в C++.

### Методические указания

Для выполнения лабораторной работы необходимо изучить следующие разделы курса лекций:

- 1.5.4 Строки.

Строка представляет собой массив символов, заканчивающийся нуль-символом. Нуль-символ — это символ с кодом, равным 0, что записывается в виде управляющей последовательности '\0’. По положению нуль-символа определяется фактическая длина строки. Строку можно инициализировать строковым литералом.

*char str[10] = "Vasia";*

В этом примере под строку выделяется 10 байт, 5 из которых занято под символы строки, а шестой — под нуль-символ. Если строка при определении инициализируется, ее размерность можно опускать (компилятор сам выделит соответствующее количество байт):

*char str[] = "Vasia";*

### Пример выполнения лабораторной работы

*Задание:*

С помощью текстового редактора создать файл, содержащий текст, длина которого не превышает 1000 символов (длина строки текста не должна превышать 70 символов). Имя файла должно иметь расширение DAT. Написать программу, которая:

- выводит текст на экран дисплея;
- определяет количество слов в тексте, у которых первый и последний символы совпадают.

*Листинг программы:*

```
#include <iostream>
#include <stdio.h>
#include <math.h>

int main()
{
	setlocale(0,"rus");
	char mas[1000], buf[50];
	int k, p;
	FILE *t;
	t = fopen("TEXT.DAT", "r");
	k = 0;
	p = 1;
	buf[p] = ' ';
	for(int i = 1; i <= 1000; i++)
	{
		while(feof(t) == 0)
		{
			fscanf(t, "%c", &mas[i]);
			printf("%c", mas[i]);
			if('A' < mas[i] && mas[i] < 'я')
			{
				buf[p] = buf[p] + mas[i];
				p++;
			}
			else
			{
				if(buf[1] == buf[p - 1] || buf[1] + 32 == buf[p])
					k++;
				buf[p] = ' ';
				p = 1;
			}
		}
	}
	printf("\nКоличество слов: %d", k);
	fclose(t);
	_gettch();
	return 0;
}
```

## Варианты заданий

Вариант 1

С помощью текстового редактора создать файл, содержащий текст, длина которого не превышает 1000 символов (длина строки текста не должна превышать 70 символов). Имя файла должно иметь расширение DAT. Написать программу, которая:

- выводит текст на экран дисплея;
- по нажатию произвольной клавиши поочередно выделяет каждое предложение текста;

- определяет количество предложений в тексте.

Вариант 2

С помощью текстового редактора создать файл, содержащий текст, длина которого не превышает 1000 символов (длина строки текста не должна превышать 70 символов). Имя файла должно иметь расширение DAT. Написать программу, которая:

- выводит текст на экран дисплея;
- по нажатию произвольной клавиши поочередно выделяет каждое слово текста;
- определяет количество слов в тексте.

Вариант 3

С помощью текстового редактора создать файл, содержащий текст, длина которого не превышает 1000 символов (длина строки текста не должна превышать 70 символов). Имя файла должно иметь расширение DAT. Написать программу, которая:

- выводит текст на экран дисплея;
- по нажатию произвольной клавиши поочередно выделяет каждое слово текста, оканчивающееся на гласную букву;
- определяет количество слов в тексте, оканчивающихся на гласную букву.

Вариант 4

С помощью текстового редактора создать файл, содержащий текст, длина которого не превышает 1000 символов. Текст должен состоять из трех предложений (длина строки текста не должна превышать 70 символов). Имя файла должно иметь расширение DAT. Написать программу, которая:

- выводит текст на экран дисплея;

- по нажатию произвольной клавиши поочередно выделяет каждое предложение текста в последовательности 2, 1, 3.

Вариант 5

С помощью текстового редактора создать файл, содержащий текст, длина которого не превышает 1000 символов (длина строки текста не должна превышать 70 символов). Имя файла должно иметь расширение DAT. Написать программу, которая:

- выводит текст на экран дисплея;
- по нажатию произвольной клавиши поочередно выделяет каждое из слов текста, у которых первый и последний символы совпадают;
- определяет количество слов в тексте, у которых первый и последний символы совпадают.

Вариант 6

С помощью текстового редактора создать файл, содержащий текст, длина которого не превышает 1000 символов (длина строки текста не должна превышать 70 символов). Имя файла должно иметь расширение DAT. Написать программу, которая:

- выводит текст на экран дисплея;
- по нажатию произвольной клавиши поочередно выделяет каждое слово текста, начинающееся на гласную букву;
- определяет количество слов в тексте, начинающихся на гласную букву.

Вариант 7

С помощью текстового редактора создать файл, содержащий текст, длина которого не превышает 1000 символов (длина строки текста не должна превышать 70 символов). Имя файла должно иметь расширение DAT. Написать программу, которая:

- выводит текст на экран дисплея;
- определяет количество символов в самом длинном слове;
- по нажатию произвольной клавиши поочередно выделяет каждое слово текста, содержащее максимальное количество символов.

Вариант 8

С помощью текстового редактора создать файл, содержащий текст, длина которого не превышает 1000 символов (длина строки текста не должна превышать 70 символов).

Имя файла должно иметь расширение DAT. Написать программу, которая:

- выводит текст на экран дисплея;
- определяет количество символов в самом коротком слове;
- по нажатию произвольной клавиши поочередно выделяет каждое слово текста, содержащее минимальное количество символов.

Вариант 9

С помощью текстового редактора создать файл, содержащий текст, длина которого не превышает 1000 символов (длина строки текста не должна превышать 70 символов).

Имя файла должно иметь расширение DAT. Написать программу, которая:

- выводит текст на экран дисплея;
- определяет в каждом предложении текста количество символов, отличных от букв и пробела;
- по нажатию произвольной клавиши поочередно выделяет каждое предложение текста, а в выделенном предложении — поочередно все символы, отличные от букв и пробела.

Вариант 10

С помощью текстового редактора создать файл, содержащий текст, длина которого не превышает 1000 символов (длина строки текста не должна превышать 70 символов).Имя файла должно иметь расширение DAT. Написать программу, которая:

- выводит текст на экран дисплея;
- определяет количество предложений текста и количество слов в каждом предложении;
- по нажатию произвольной клавиши поочередно выделяет каждое предложение текста, а в выделенном предложении — поочередно все слова.

Вариант 11

С помощью текстового редактора создать файл, содержащий текст, длина которого не превышает 1000 символов (длина строки текста не должна превышать 70 символов).

Имя файла должно иметь расширение DAT. Написать программу, которая:

- выводит текст на экран дисплея;
- определяет количество букв 'a' в последнем слове текста;
- по нажатию произвольной клавиши выделяет последнее слово текста, а в выделенном слове поочередно все буквы 'a'.

Вариант 12

С помощью текстового редактора создать файл, содержащий текст, длина которого не превышает 1000 символов (длина строки текста не должна превышать 70 символов). Имя файла должно иметь расширение DAT. Написать программу, которая:

- выводит текст на экран дисплея;
- определяет самую длинную последовательность цифр в тексте (считать, что любое количество пробелов между двумя цифрами не прерывает последовательности цифр);
- по нажатию произвольной клавиши поочередно выделяет каждую последовательность цифр, содержащую максимальное количество символов.

Вариант 13

С помощью текстового редактора создать файл, содержащий текст, длина которого не превышает 1000 символов (длина строки текста не должна превышать 70 символов). Имя файла должно иметь расширение DAT. Написать программу, которая:

- выводит текст на экран дисплея;

- определяет порядковый номер заданного слова в каждом предложении текста (заданное слово вводится с клавиатуры);
- по нажатию произвольной клавиши поочередно выделяет каждое предложение текста, а в выделенном предложении — заданное слово.

Вариант 14

С помощью текстового редактора создать файл, содержащий текст, длина которого не превышает 700 символов (длина строки текста не должна превышать 70 символов).

Имя файла должно иметь расширение DAT. Написать программу, которая:

- выводит текст на экран дисплея;
- по нажатию произвольной клавиши поочередно выделяет в тексте заданное слово (заданное слово вводить с клавиатуры);
- выводит текст на экран дисплея еще раз, выкидывая из него заданное слово и удаляя лишние пробелы.

Вариант 15

С помощью текстового редактора создать файл, содержащий текст, длина которого не превышает 700 символов (длина строки текста не должна превышать 70 символов).

Имя файла должно иметь расширение DAT. Написать программу, которая:

- выводит текст на экран дисплея;
- по нажатию произвольной клавиши поочередно выделяет в тексте заданные слова, которые нужно поменять местами (заданные слова вводить с клавиатуры);
- выводит текст на экран дисплея еще раз, меняя в нем местами заданные слова и удаляя лишние пробелы.

Вариант 16

С помощью текстового редактора создать файл, содержащий текст, длина которого не превышает 700 символов (длина строки текста не должна превышать 70 символов). Имя файла должно иметь расширение DAT. Написать программу, которая:

- выводит исходный текст на экран дисплея;
- по нажатию произвольной клавиши поочередно выделяет в тексте заданное слово (заданное слово вводить с клавиатуры);
- выводит текст на экран дисплея еще раз, заключая заданное слово в кавычки, и поочередно выделяет заданное слово вместе с кавычками.

Вариант 17

С помощью текстового редактора создать файл, содержащий текст, длина которого не превышает 700 символов (длина строки текста не должна превышать 70 символов). Имя файла должно иметь расширение DAT. Написать программу, которая:

- выводит исходный текст на экран дисплея;
- выводит текст на экран дисплея еще раз, вставляя в каждое предложение в качестве последнего заданное слово, введенное с клавиатуры в качестве исходных данных;
- по нажатию произвольной клавиши поочередно выделяет в тексте вставленное слово.

Вариант 18

С помощью текстового редактора создать файл, содержащий текст, длина которого не превышает 500 символов (длина строки текста не должна превышать 70 символов). Имя файла должно иметь расширение DAT. Написать программу, которая:

- выводит текст на экран дисплея;

- по нажатию произвольной клавиши поочередно выделяет в тексте лишние пробелы между словами;
- выводит текст на экран дисплея еще раз, убирая лишние пробелы между словами и начиная каждое предложение с новой строки.

Вариант 19

С помощью текстового редактора создать файл, содержащий текст, длина которого не превышает 700 символов (длина строки текста не должна превышать 70 символов). Имя файла должно иметь расширение DAT. Написать программу, которая:

- выводит текст на экран дисплея;
- по нажатию произвольной клавиши поочередно выделяет в тексте заданное слово (заданное слово вводить с клавиатуры); выводит текст на экран дисплея еще раз, заменяя в заданном слове строчные буквы прописными.

Вариант 20

С помощью текстового редактора создать файл, содержащий текст, длина которого не превышает 1000 символов (длина строки текста не должна превышать 70 символов). Имя файла должно иметь расширение DAT. Написать программу, которая:

- выводит текст на экран дисплея; определяет наибольшее количество подряд идущих пробелов в тексте;

по нажатию произвольной клавиши поочередно выделяет каждую из последовательностей пробелов максимальной длины.