**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ**
**Федеральное государственное автономное образовательное учреждение высшего образования**
**«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»**
**Невинномысский технологический институт (филиал)**

# Методические указания по выполнению лабораторных работ по дисциплине «Программирование и алгоритмизация»

**Направление подготовки 15.03.04 Автоматизация технологических процессов и производств**
**Квалификация выпускника – бакалавр**

Невинномысск, 2019

# Содержание

## Введение

Лабораторный практикум содержит информационный материал, необходимый бакалаврам направления подготовки 15.03.04 Автоматизация технологических процессов и производств для выполнения лабораторных работ по дисциплине “Программирование и алгоритмизация”

Лабораторные занятия проводятся с целью приобретения практических навыков алгоритмизации, программирования, тестирования и отладки программ на компьютере с использованием современных технологий и инструментальных средств.

Перечень лабораторных работ:
- Лабораторная работа №1. Управляющая структура «Следование».
- Лабораторная работа №2. Управляющая структура «Развилка».
- Лабораторная работа №3. Управляющая структура «Выбор».
- Лабораторная работа №4. Управляющие структуры «Циклы».
- Лабораторная работа №5. Суммирование рядов.
- Лабораторная работа №6. Обработка массивов.
- Лабораторная работа №7. Методы сортировки.
- Лабораторная работа №8. Обработка строк.
- Лабораторная работа №9. Текстовые файлы.
- Лабораторная работа №10. Базы данных.
- Лабораторная работа №11. Линейные списки.
- Лабораторная работа №12. Динамические структуры данных.
- Лабораторная работа №13. Классы. Объекты.

## Лабораторная работа №1
## Управляющая структура “Следование”

***Цель лабораторной работы:*** *изучение концепций и освоение технологии структурного программирования, приобретение навыков структурного программирования на языке С/С++ при решении простейших вычислительных задач.*

***Задание на программирование:*** *используя технологию структурного программирования, разработать линейную программу решения индивидуальной вычислительной задачи.*

***Порядок выполнения работы:***

1) Получить у преподавателя индивидуальное задание и выполнить *постановку задачи*: сформулировать условие, определить входные и выходные данные.
2) Разработать *математическую модель* вычислений.
3) Построить *схему алгоритма* решения задачи.
4) Составить программу на языке С/С++.
5) В программе использовать данные типа ***unsigned char***.
5) *Выходные данные (сообщения)* выводить на экран в развернутой форме.
6) Проверить и продемонстрировать преподавателю работу программы.
7) Оформить *отчет о лабораторной работе* в составе: *постановка задачи, математическая модель, схема алгоритма решения, текст программы, контрольные примеры.*

**Варианты индивидуальных заданий**

**Выполнить поразрадные логические операции над машинными кодами**

**1**
117 **AND** 90
-117 **XOR** 90
117 → 3
**NOT** 21 **XOR** –13 **AND** (-23 **OR NOT** 9)

**2**
135 **AND** 106
135 **OR** -106
135 → 4
**NOT** 17 **OR** (**NOT** 111 **XOR** –19) **AND** 91

**3**
207 **AND** 37
207 **XOR** -37
37 ← 2
-21 **AND** (**NOT** 75 **OR** –20) **XOR NOT** 59

**4**
27 **AND** 13
-27 **OR** 13
27 <- 2
**NOT** 21 **XOR** –3 **AND** (**NOT** 26 **OR** –13)

**5**
-21 **OR** 43
21 **XOR** 43
43 ← 2
(**NOT** 19 **OR** –6) **AND NOT** –9 **XOR** 4

**6**
55 **AND** 15
55 **XOR** -15
15 ← 3
**NOT** 7 **AND** –5 **XOR** (**NOT** 127 **OR** –8)
→←
**7**
99 OR -17
99 AND 17
17 ← 2
(18 **OR NOT** –8) **AND NOT** –7 **XOR** 3

**8**
29 **OR** -49
29 **XOR** 49
49 ← 4
(**NOT** 8 **XOR** –6) **AND** 9 **XOR NOT** -12

**9**
42 **AND** 17
42 **OR** -17
42 → 3
**NOT** 25 **XOR** –4 **AND** (**NOT** 22 **OR** –10)

**10**
36 **AND** 12
36 **XOR** 12
36 ← 3
**NOT** –3 **XOR** 15 **AND** (**NOT** 8 **OR** –6)

**11**
25 **AND** 18
25 **XOR** 18
25 ← 2
**NOT** 23 **OR** –4 **AND** (**NOT** 24 **OR** –9)

**12**
39 **AND** 14
39 **OR** -14
39 ← 3
**NOT** 17 **AND** –5 **OR** (25 **AND NOT** –9)

**13**
49 AND 11
49 XOR 11
49 → 2
15 **OR NOT** –3 **AND** (14 **OR NOT** 16)

**14**
180 **AND** 35
180 **XOR** 35
35 ← 2
**NOT** –7 **OR** 8 **AND** (26 **XOR NOT** –9)

**15**
120 **AND** 37
120 **OR** -37
120 → 2
85 **OR NOT** –9 **AND** (**NOT** 46 **OR** –13)

**16**
137 **AND** 80
137 **XOR** 80
137 → 3
105 **XOR NOT** –15 **AND** (**NOT** 82 **OR** –25)

**17**
157 **AND** 14
157 **XOR** 14
157 → 4
110 **OR NOT** –25 **AND** (**NOT** 46 **XOR** –11)

**18**
139 **AND** 18
139 **OR** -18
139 → 3
80 **OR NOT** –11 **AND** (**NOT** 48 **XOR** –15)

**19**
125 **AND** 20
125 **OR** -20
125 ← 1
40 **OR NOT** –19 **AND** (**NOT** 50 **XOR** –7)

**20**
94 **AND** 15
94 **XOR** 15
94 → 2
86 **XOR NOT** –17 **AND** (**NOT** 40 **OR** –9)

**21**
102 **AND** 31
102 **OR** -31
102 → 3
35 **XOR NOT** –9 **AND** (**NOT** 28 **OR** –17)

**22**
90 **AND** 11
90 **OR** -11
90 ← 2
17 **XOR NOT** –11 **AND** (**NOT** 30 **OR** –15)

**23**
74 **AND** 111
74 **XOR** 111
74 ← 3
28 **OR NOT** –13 **AND** (**NOT** 16 **XOR** –25)

**24**
36 **AND** 21
36 **XOR** 21
36 ← 4
14 **OR NOT** –15 **AND** (**NOT** 26 **XOR** –17)

**25**
61 **AND** 18
61 **OR** -18
61 ← 4
9 **XOR NOT** –21 **AND** (**NOT** 60 **OR** –5)

**26**
75 AND 26
75 XOR 26
72 ← 4
**NOT** 80 **XOR** –31 **AND** (-16 **OR NOT** 11)

**27**
81 **AND** 14
81 **XOR** 14
81 ← 3
70 **XOR NOT** –11 **AND** (**NOT** 36 **OR** 15)

**28**
111 **AND** 14
111 **XOR** 14
111 ← 3
15 **XOR NOT** –9 **AND** (**NOT** 26 **OR** 31)

***Пример программы***

```
#include<stdio.h>
#include<conio.h>
void main()
{unsigned char a, b, c;
 clrscr();
 a = 41 & -21;
 printf("41 AND -21 = (41) = %i\n", a);
 a = -41 & -21;
 printf("-41 AND -21 = (195) = %i\n", a);
 b = 41 | 21;
 printf("41 OR 21 = (61) = %i\n", b);
 b = 41 ^ 21;
 printf("41 XOR 21 = (60) = %i\n", b);
 b = 41 << 2;
 printf("41 << 2 = (164) = %i\n", b);
 c = ~43 | -9 & (~-7 ^ 4);
 printf("NOT 43 OR -9 AND (NOT-7 XOR 4) = (214) = %i\n", c);
 getchar();

 a = 141 & -121;
 printf("141 AND -121 = (133) = %i\n", a);
 a = -141 & -121;
 printf("-141 AND -121 = (3) = %i\n", a);
 b = 141 | 121;
 printf("141 OR 121 = (253) = %i\n", b);
 b = 141 ^ 121;
 printf("141 XOR 121 = (244) = %i\n", b);
 b = -1 >> 2;
 printf("-1 >> 2 = (255) = %i\n", b);
 getchar();

 a = 111 & -12;
 printf("111 AND -12 = (100) = %i\n", a);
 a = -111 & -12;
 printf("-111 AND -12 = (144) = %i\n", a);
 b = 111 | 12;
 printf("111 OR 12 = (111) = %i\n", b);
 b = 111 ^ 12;
 printf("111 XOR 12 = (99) = %i\n", b);
 b = 111 >> 2;
 printf("111 >> 2 = (27) = %i\n", b);
 getchar();
}
```

## Лабораторная работа №2
## Управляющая структура “Развилка”

***Цель лабораторной работы:*** *изучение концепций и освоение технологии структурного программирования, приобретение навыков структурного программирования на языке С/С++ при решении логических задач.*

***Задание на программирование:*** *используя технологию структурного программирования, разработать разветвляющуюся программу для решения индивидуальной задачи определения места нахождения на плоскости точки с произвольно заданными координатами.*

***Порядок выполнения работы:***

1) Получить у преподавателя индивидуальное задание и выполнить *постановку задачи*: сформулировать условие, определить входные и выходные данные.

2) Разработать *математическую модель*: привести уравнения линий, ограничивающих выделенные штриховкой области, описать условия попадания точки в каждую область (количество областей должно быть от 3 до 6).

3) Построить *схему алгоритма* решения задачи.

4) Составить программу на языке С/С++.

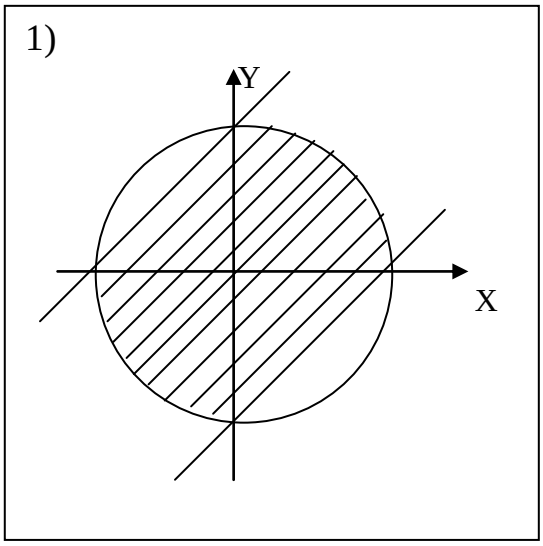
5) *Входные данные* вещественного типа ***float*** вводить с клавиатуры по запросу. *Выходные данные (сообщения)* выводить на экран в развернутой форме.

6) Проверить и продемонстрировать преподавателю работу программы на *полном* наборе тестов.

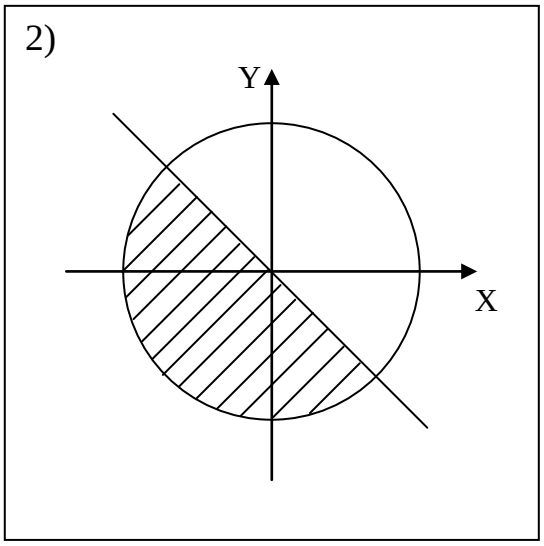
7) Оформить *отчет о лабораторной работе* в составе: *постановка задачи, математическая модель, схема алгоритма решения, текст программы, контрольные примеры.*

**Варианты индивидуальных заданий**

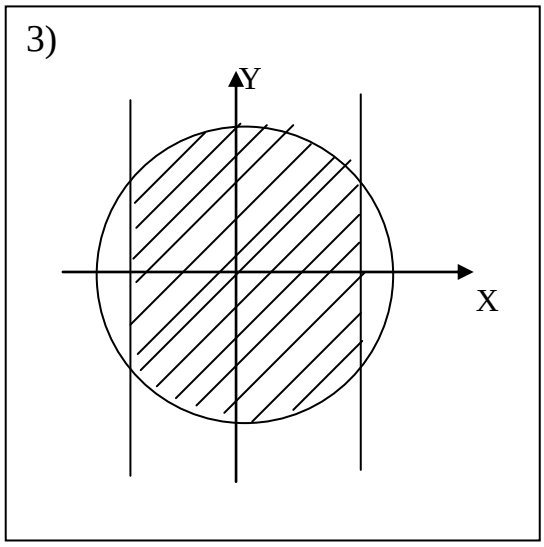
1)



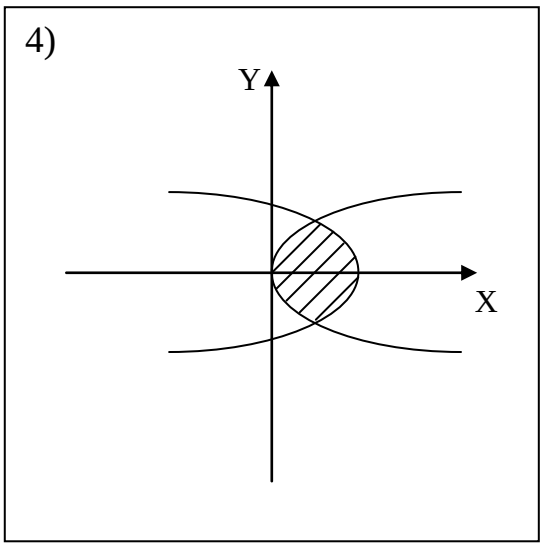

2)



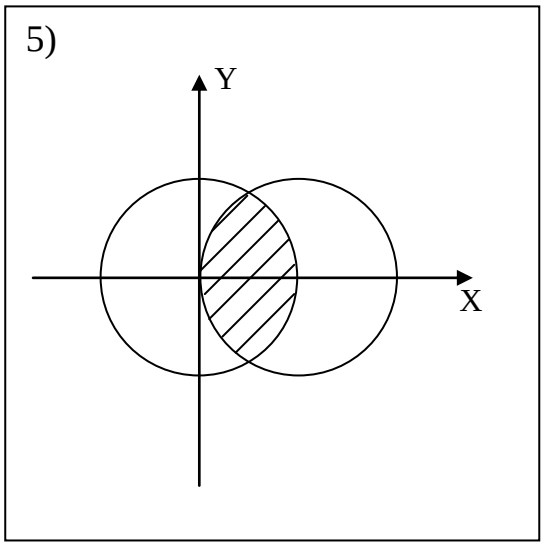

3)



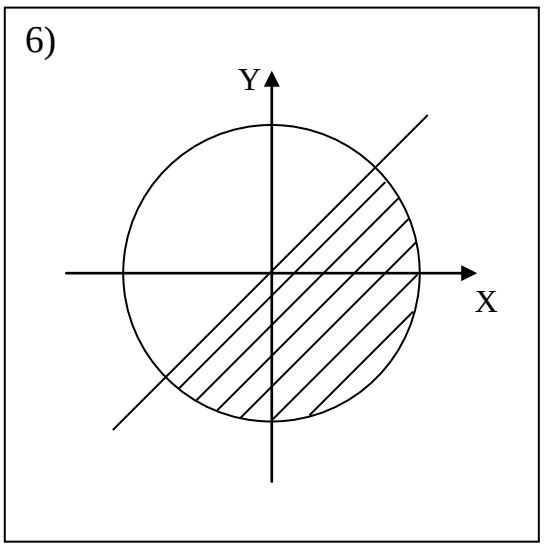

4)



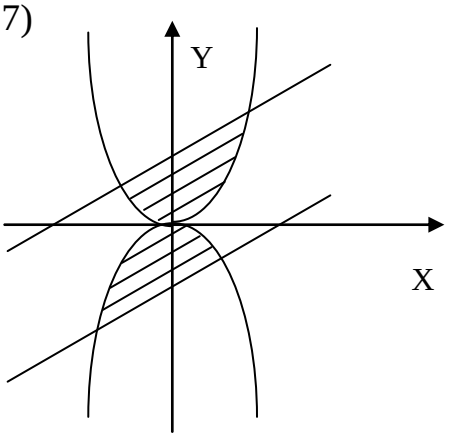

5)



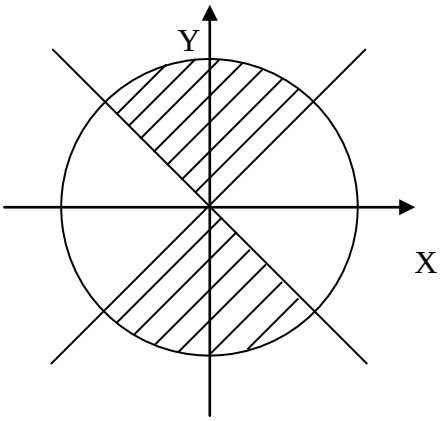

6)



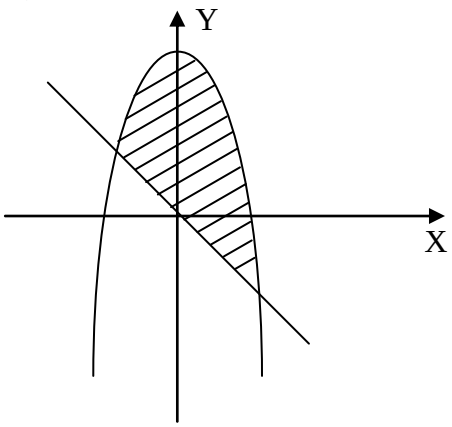
7)



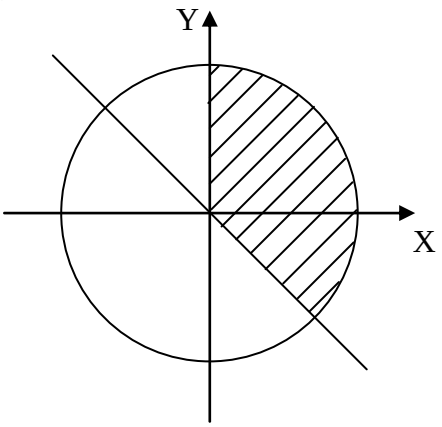

8)



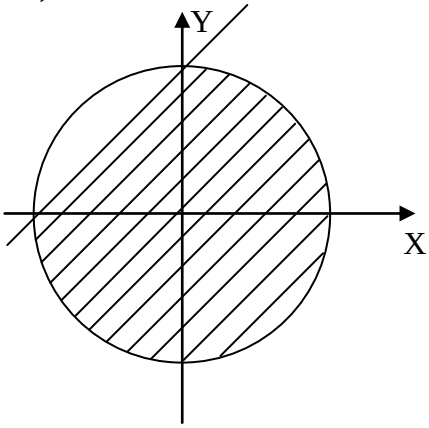

9)



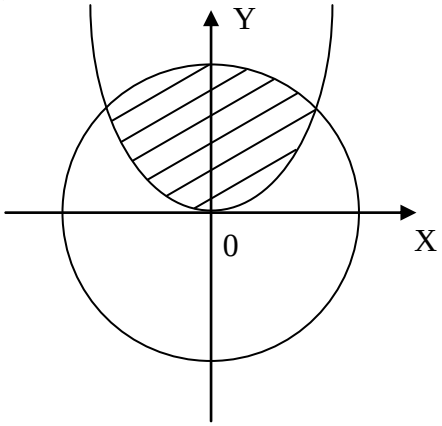

10)



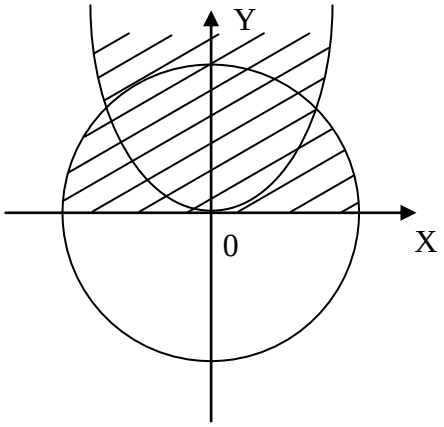

11)



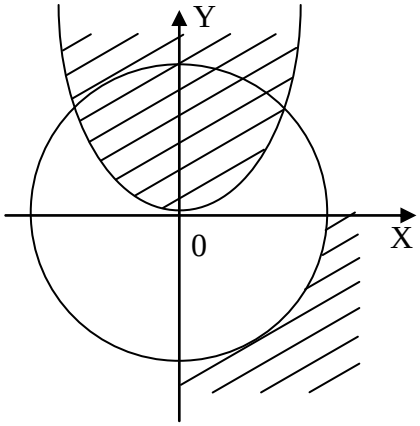
12)



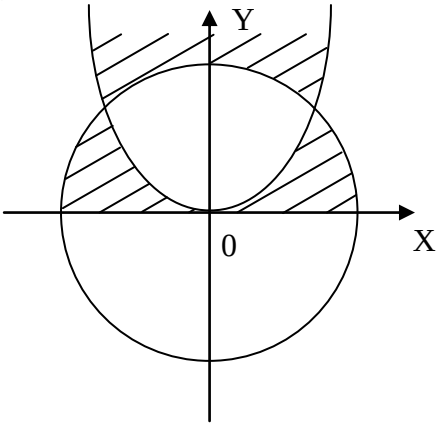

13)



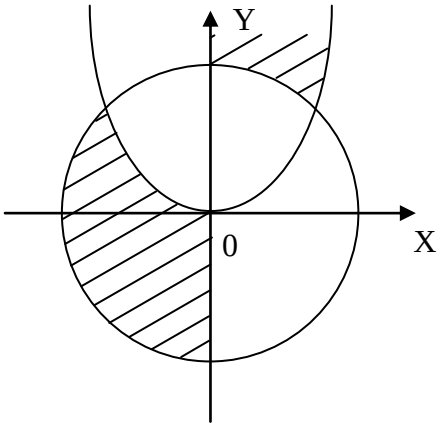

14)



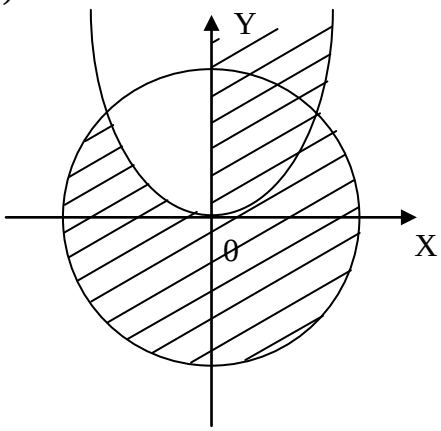

15)



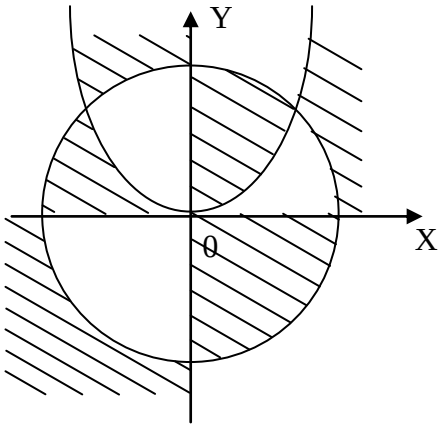

16)



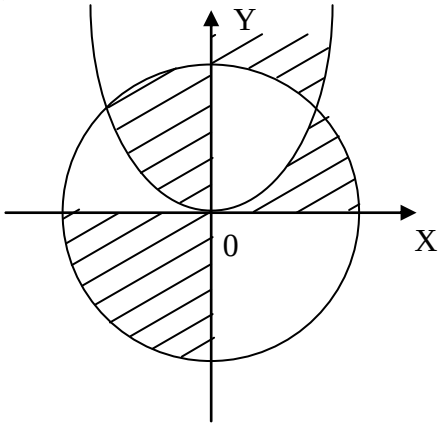

17)



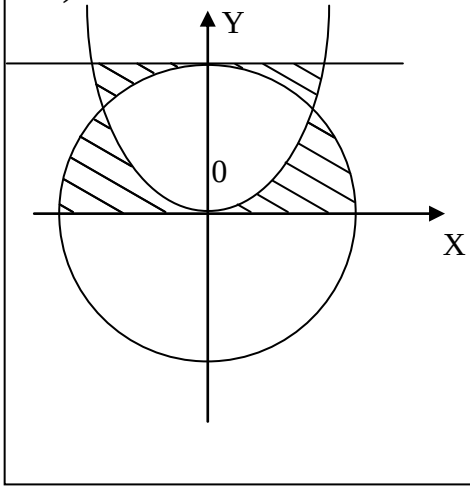
18)



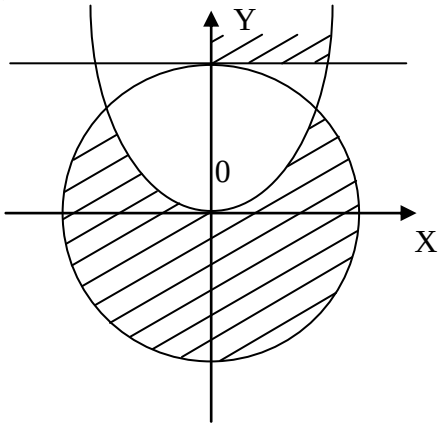

19)



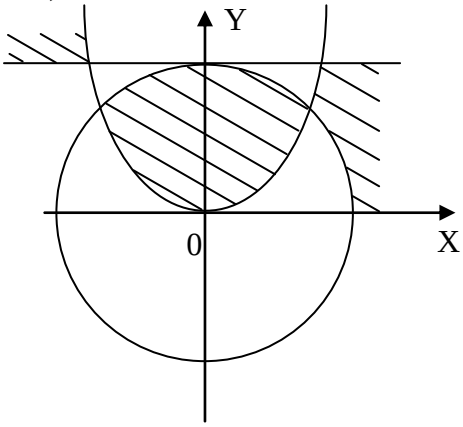

20)



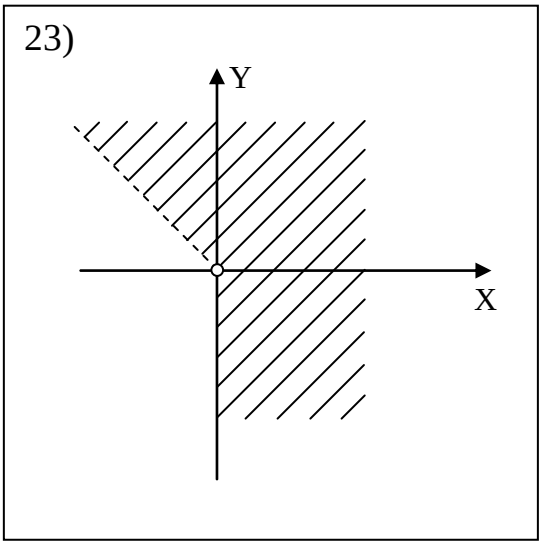

21)



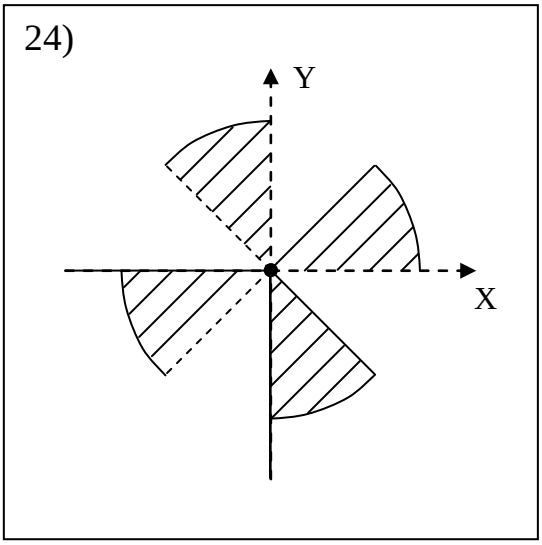

22)



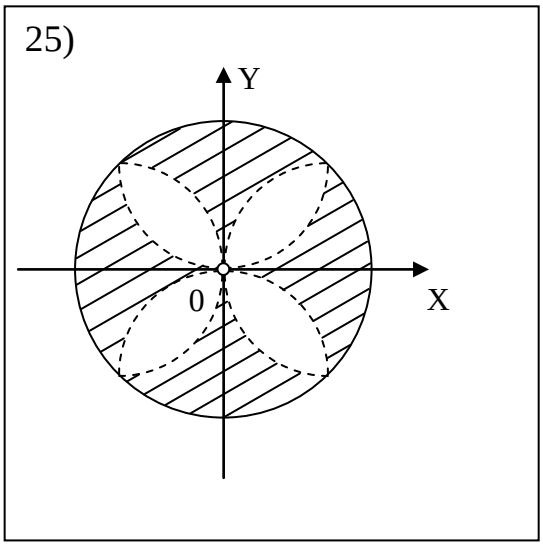
23)



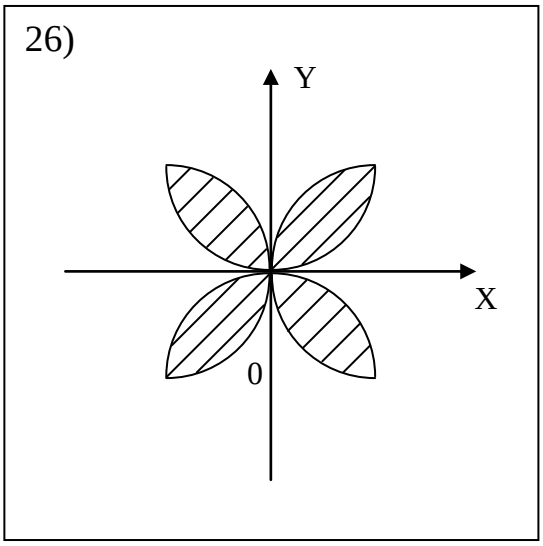

24)



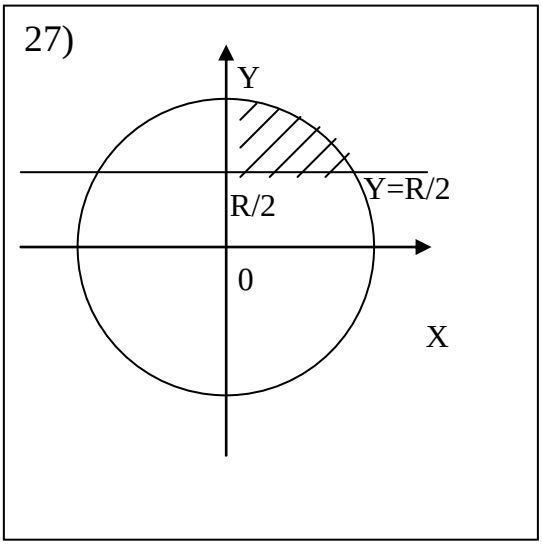

25)



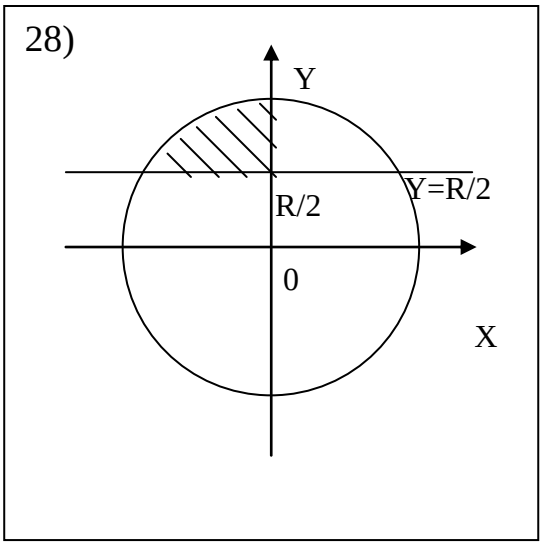

26)



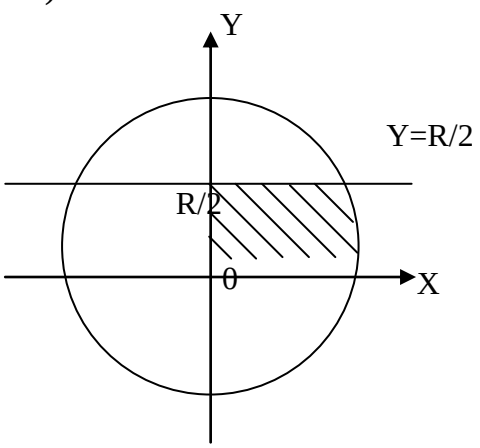

27)



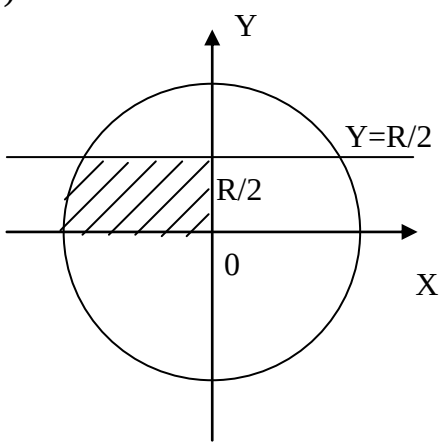

28)



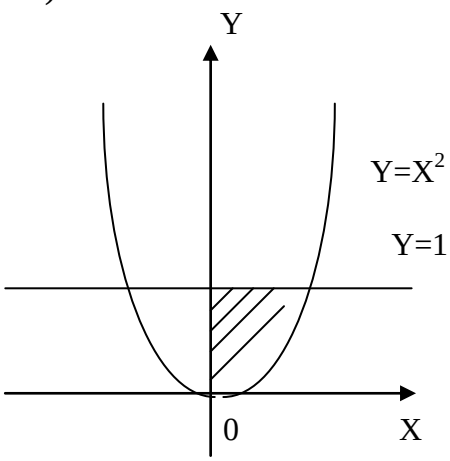
29)



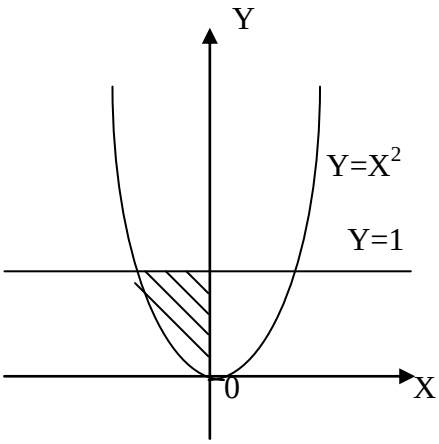

30)



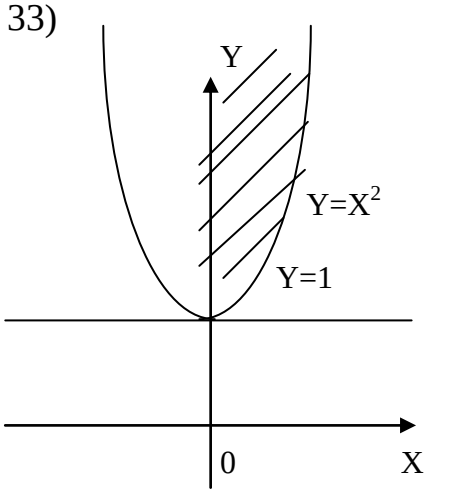

31)



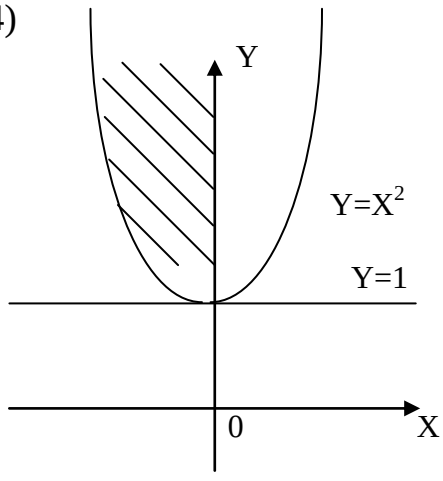

32)



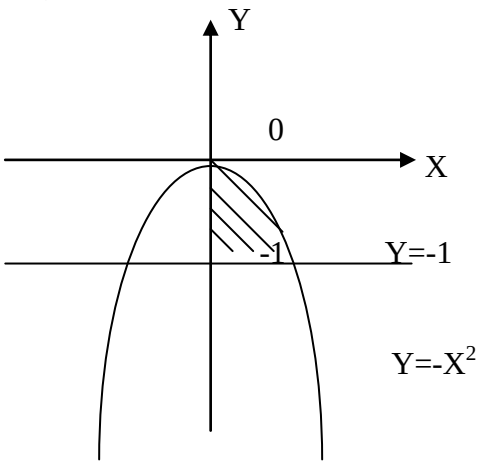

33)



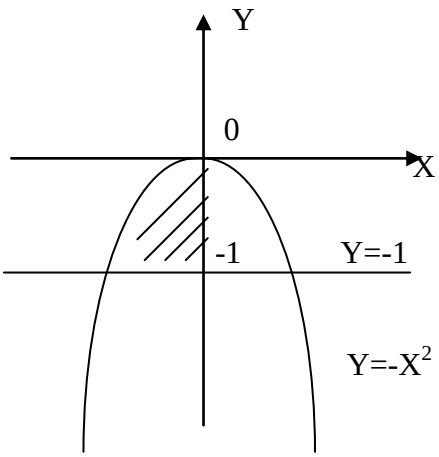

34)



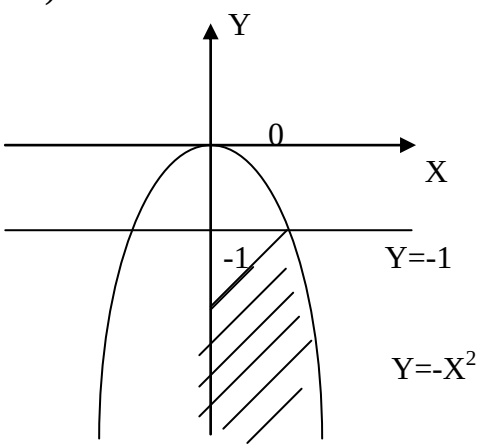
35)



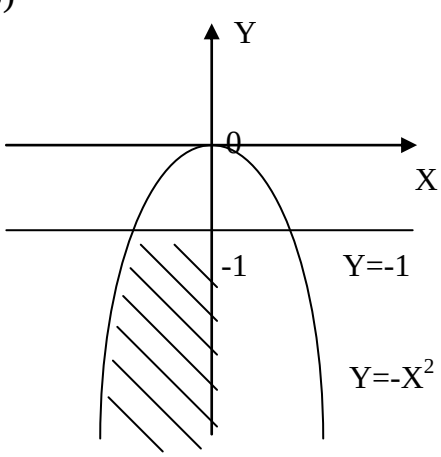

36)



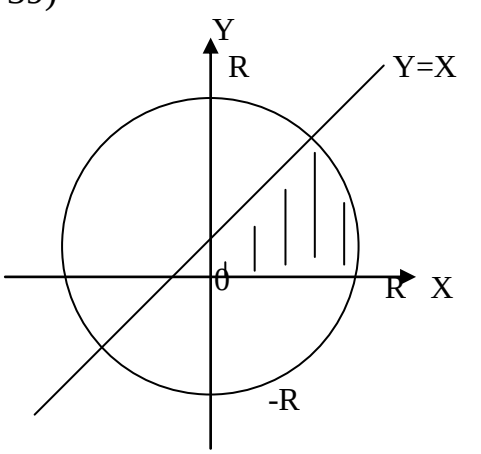

37)



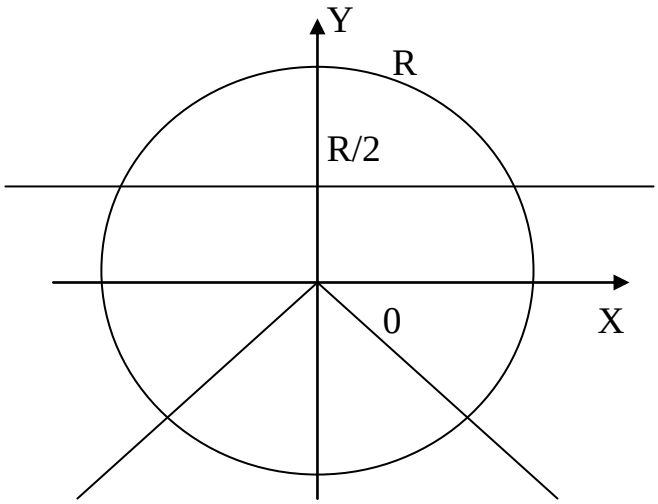

38)

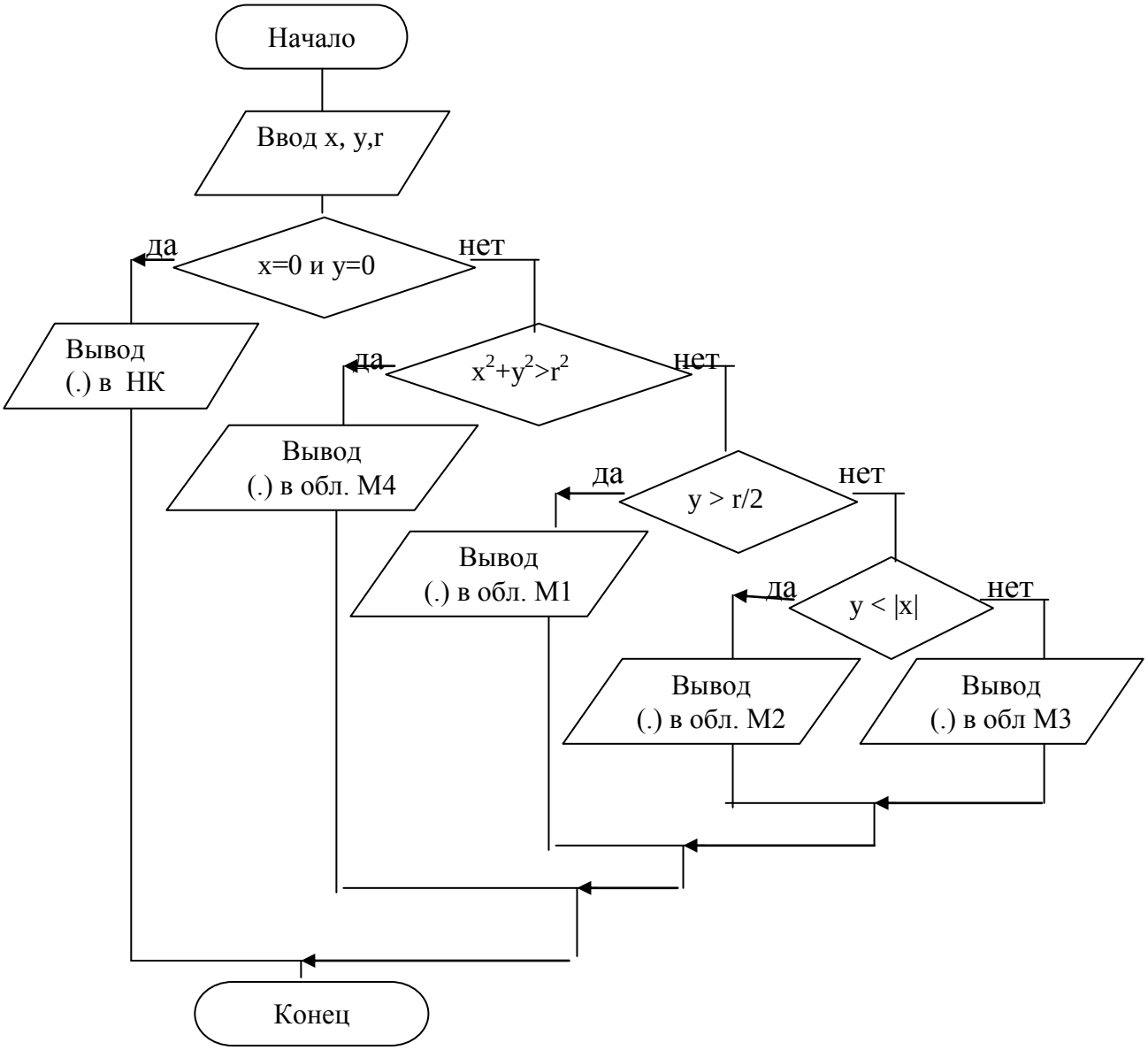



39)



***Пример схемы алгоритма и текста программы определения местоположения точки для варианта задания вида:***



```
//Пример решения
#include<iostream.h>
#include<conio.h>
#include<math.h>

#include<iostream.h>
#include<conio.h>
#include<math.h>

int main()
{int i;
 float x, y,                  //координаты точки
         r;                   //радиус окружности

 clrscr();
 cout << "Введите координаты и радиус: x,y,r \n";
 cin >> x >> y >> r;
 if(x == 0 && y == 0) cout << "Точка в начале координат\n";
 else if(x * x + y * y > r * r) cout << "Точка в области М4\n";
      else if(y > r / 2) cout << "Точка в области М1\n";
          else if(y < fabs(x)) cout << "Точка в области М2\n";
               else cout << "Точка в области М3\n";
 cout << "\n Повторить-1, Выход-2: ";
 cin >> i;
 if (i == 1) main();
 return 0;
}
```

## Лабораторная работа №3
## Управляющая структура “Выбор”

***Цель лабораторной работы:*** *изучение концепций и освоение технологии структурного программирования, приобретение навыков структурного программирования на языке С/С++ многовариантных вычислений.*

***Задание на программирование:*** *используя технологию структурного программирования, разработать разветвляющуюся программу для решения индивидуальной задачи выбора варианта вычисления по ключу с использованием оператора-переключателя switch.*

***Порядок выполнения работы:***

1) Получить у преподавателя индивидуальное задание и выполнить *постановку задачи*: сформулировать условие, определить входные и выходные данные.

2) Разработать *математическую модель*:

- составить список различных вариантов получения выходных данных задачи;
- выявить ключ выбора - данное целого типа, значения которого могут служить ключами различных вариантов выполнения действий;
- с помощью формул описать варианты получения выходных данных задачи в зависимости от значения ключа выбора варианта.

3) Построить *схему алгоритма* решения задачи.

4) Составить программу на языке С/С++.

5) *Входные данные* вводить с клавиатуры по запросу.

*Выходные данные* выводить на экран в развернутой форме с пояснениями.

6) Проверить и продемонстрировать преподавателю работу программы на *полном* наборе тестов, в том числе с ошибочными входными данными.

7) Оформить *отчет о лабораторной работе* в составе: *постановка задачи, математическая модель, схема алгоритма решения, текст программы, контрольные примеры.*

## Варианты индивидуальных заданий

**1**
Определить название месяца года, следующего за заданным месяцем.

**2**
Определить название k-го месяца после заданного месяца года.

**3**
Определить название столицы по заданному названию страны.

**4**
Определить название десятичной цифры по заданному ее значению.

**5**
Определить написание заданной десятичной цифры римскими цифрами.

**6**
Определить двоичный код заданной десятичной цифры.

**7**
Определить сезон года (зима, весна, лето, осень), на который приходится заданный месяц.

**8**
Определить название континента (Азия, Америка, Африка, Европа) по заданному названию страны.

**9**
Определить название цвета радуги, следующего за заданным цветом.

**10**
Определить название интервала (секунда, терция, кварта, квинта, секста, септима), образованного двумя заданными нотами (до, ре, ми, фа, соль, ля, си).

**11**
Определить величину в метрах некоторой длины, заданной в одной из указанных единиц измерения (километр, метр, дециметр, сантиметр, миллиметр).

**12**
Для целого числа k от 1 до 130 вывести фразу “Мне k лет”, учитывая при этом, что при некоторых значениях k слово “лет” надо заменить словом “год” или “года”.

**13**
Для натурального числа k вывести фразу “Мы нашли k грибов в лесу”, согласовав слово “гриб” с числом k.

**14**
Для целого числа d от 1 до 9999, обозначающего денежную единицу, дописать слово “рубль” в правильной форме.

**15**
Для целого числа d от 1 до 9999, обозначающего денежную единицу, дописать слово “копейка” в правильной форме.

**16**
Вычислить стоимость междугородного телефонного разговора заданной продолжительности. Цена одной минуты определяется по указанному коду города.

**17**
Вывести указанное слово из группы однотипно склоняемых слов (степь, боль, тетрадь, дверь) в заданном падеже (им., род., дат., вин., твор., предл.).

**18**
Корабль сначала шел по заданному курсу (север, восток, юг, запад). Затем его курс был изменен согласно заданному приказу (вперед, вправо, назад, влево). Определить новый курс корабля.

**19**
Определить количество дней в указанном месяце заданного года.

**20**
Определить, образует ли заданная тройка чисел y (год), m (месяц), d (день) правильную дату.

**21**
По заданной дате d (день), m (месяц), y(год) определить дату d1, m1, y1 следующего дня.

**22**
Определить порядковый номер того дня високосного года, который имеет заданную дату d (день), m (месяц).

**23**
Определить d (день), m (месяц) – дату k-го по счету дня високосного года.

**24**

Считая, что год не високосный и его 1 января приходится на день недели wd1, определить wd – день недели, на который приходится день с датой d (день), m (месяц).

**25**

Считая, что год не високосный и его 1 января приходится на день недели wd1, определить количество пятниц в году, приходящихся на 13-е числа месяца.

### *Пример программы с оператором switch*

```
//Вычисление площадей геометрических фигур.
//Входные данные:  t - тип фигуры,
//           a, h, r - параметры фигур.
//Выходные данные: s - площадь фигуры.
#include<iostream.h>
#include<conio.h>
#include<math.h>

int main()
{int i, t;
 float a, h, r, s;

 clrscr();
 cout << "Задайте тип фигуры:\n";
 cout << "1 - квадрат, 2 - прямоугольник, 3 - круг -> ";
 cin >> t;
 if(t < 1 || t > 3) cout << "\nОшибочный тип фигуры!!!";
 else {switch(t)
          {case 1: cout << "Введите длину стороны квадрата: ";
                   cin >> a;
                   s = a * a;
                   break;
           case 2: cout << "Введите размеры сторон прямоугольника: ";
                   cin >> a >> h;
                   s = a * h;
                   break;
           case 3: cout << "Введите радиус круга: ";
                   cin >> r;
                   s = M_PI * r * r;
          }
        cout << "Площадь фигуры равна: " << s;
       }
 cout << "\n Повторить-1, Выход-2: ";
 cin >> i;
 if (i == 1) main();
 return 0;
}
```

## Лабораторная работа №4
## Управляющие структуры “Циклы”

***Цель лабораторной работы:*** *изучение концепций и освоение технологии структурного программирования, приобретение навыков структурного программирования на языке С/С++ циклических вычислений.*

***Задание на программирование:*** *используя технологию структурного программирования, разработать программу решения двух индивидуальных задач, содержащую 3 вида циклических управляющих структур: Цикл - Пока (с предусловием), Цикл - До (с постусловием), Цикл - Для (с параметром). Реализовать интерфейс, обеспечивающий заданное расположение и назначение окон на экране при выполнении программы в соответствии с индивидуальным заданием*

***Порядок выполнения работы:***

1) Получить у преподавателя индивидуальное задание: а) схему расположения и назначения окон на экране; б) две индивидуальные задачи. Выполнить *постановку двух задач*: сформулировать условие, определить входные и выходные данные.

2) Разработать *математическую модель*.

3) Построить *схему алгоритма*, используя для решения каждой из задач все три циклические управляющие структуры (операторы ***while***, ***do…while***, ***for***).

4) Составить программу на языке С/С++.

5) *Входные данные* вводить с клавиатуры по запросу.

*Выходные данные* выводить на экран в развернутой форме с пояснениями.

6) Проверить и продемонстрировать преподавателю работу программы на *полном* наборе тестов, в том числе с ошибочными входными данными.

7) Оформить *отчет о лабораторной работе* в составе: *постановка задачи, математическая модель, схема алгоритма решения, текст программы, контрольные примеры.*

## *Варианты индивидуальных заданий*

**1**

1. По введенным с клавиатуры значениям X, m вычислить S:

$$S = \sum_{i=1,3,5,\ldots}^{2\cdot m+1} i \cdot X^{-2}$$

2. Вычислить предел последовательности $\{Y_n\}$ при $n\to\infty$, где $Y_n$ вычисляется по формуле $Y_n = 0{,}25 * \sin(Y_{n-1}) + \sin(Y_{n-2})$; n = 2,3,4,…

Значения $Y_0$, $Y_1$ вводятся с клавиатуры. Вычисления прекратить при выполнении условия $|Y_n - Y_{n-1}|<\varepsilon$.

**2**

1. По введенным с клавиатуры значениям X и m вычислить P:

$$P = \prod_{i=1}^{m}(m + \frac{X}{m-i+1})$$

2. Вычислить предел последовательности $\{Y_n\}$ при $n\to\infty$, где $Y_n$ вычисляется по формуле $Y_n= 0.2 + 0.1 \sin(Y_{n-1})$; n=1,2,3...

Значение $Y_0$ вводится с клавиатуры. Вычисления прекратить при выполнении условия $|Y_n – Y_{n-1}|<\varepsilon$.

**3**

1. По введенным с клавиатуры значениям A, B, n, m и X вычислить S:

$$S = A + \sum_{i=m}^{n}(X + \frac{B}{i})^2$$

2. Вычислить предел последовательности $\{Y_n\}$ при $n\to\infty$, где $Y_n$ вычисляется по формуле

$Y_n=0.1 \operatorname{tg}(Y_{n-1}) + 0.3 \operatorname{tg}(Y_{n-3})$; n=3,4,5...

Значения $Y_0$, $Y_1$, $Y_2$ вводятся с клавиатуры. Вычисления прекращаются при выполнении условия $|Y_n – Y_{n-1}|<\varepsilon$.

**4**

1. По введенным с клавиатуры значениям A, B, n и X вычислить S:

$$S = A + B \sum_{i=2,4,6,\ldots}^{n} \frac{X - A\cdot B\cdot i}{X + A\cdot B\cdot i}$$

2. Вычислить предел последовательности $\{Y_n\}$ при $n\to\infty$, где $Y_0=0$, а $Y_n$ вычисляется по формуле $Y_n = \frac{1}{1+Y_{n-1}}; n = 1,2,3...$

Значение $Y_0$ вводится с клавиатуры. Вычисления прекращаются при выполнении условия $|Y_n – Y_{n-1}|<\varepsilon$.

**5**

1. По введенным с клавиатуры значениям A, B, n, m и X вычислить S:

$$S = A + B\sum_{i=m}^{n}(-1)^i \frac{A + X \cdot i}{B + X \cdot i}$$

2. Вычислить предел последовательности $\{Y_n\}$ при $n \to \infty$, где $Y_n$ вычисляется по формуле $Y_n = 0{,}352 * Y_{n-1} + \cos(\pi/2 + Y_{n-2})$; n = 2,3,4…

Значения $Y_0$, $Y_1$ вводятся с клавиатуры. Вычисления прекратить при выполнении условия $|Y_n - Y_{n-1}| < \varepsilon$.

**6**

1. Вычислить сумму S значений функции Y=f(x):

$$S = \sum_i \frac{x^2 - 3x + 2}{\sqrt{2x^2 - 1}}; \text{ при } x = 1.5 + 0.1 \cdot i; i = \overline{1,40}$$

2. Вычислить предел последовательности $\{Y_n\}$ при $n \to \infty$, где $Y_n$ вычисляется по формуле

$$Y_n = \frac{1}{\sqrt{12 + Y_{n-1}^2 + Y_{n-2}^2}}; n = 2,3,4...$$

Значения $Y_0$, $Y_1$ вводятся с клавиатуры. Вычисления прекратить при выполнении условия $|Y_n - Y_{n-1}| < \varepsilon$.

**7**

1. Вычислить сумму S значений функции Y=f(x):

$$S = \sum_i \lg\left(\frac{x^2 + 1}{(i-1)!}\right); \text{ при } x = -1 + 0.2 \cdot i; i = \overline{1,10}$$

2. Вычислить предел последовательности $\{Y_n\}$ при $n \to \infty$, где $Y_n$ вычисляется по формуле

$$Y_n = \frac{1}{\sqrt{10 + Y_{n-2}^2 + Y_{n-3}^2}}; n = 3,4,5...$$

Значения $Y_0$, $Y_1$, $Y_2$ вводятся с клавиатуры. Вычисления прекратить при выполнении условия $|Y_n - Y_{n-1}| < \varepsilon$.

**8**

1. По введенному с клавиатуры значению X вычислить S:

$$S = \frac{(X-2)\cdot(X-4)\cdot(X-8)\cdot\ldots\cdot(X-128)}{(X-1)\cdot(X-3)\cdot(X-7)\cdot\ldots\cdot(X-127)}$$

2. Вычислить предел последовательности $\{Y_n\}$ при $n\to\infty$, где $Y_n$ вычисляется по формуле

$$Y_n = \frac{1}{\sqrt{1+\sin^2 Y_{n-1}+\sin^2 Y_{n-2}}}; n = 2,3,4\ldots$$

Значения $Y_0$, $Y_1$ вводятся с клавиатуры. Вычисления прекратить при выполнении условия $|Y_n - Y_{n-1}|<\varepsilon$.

**9**

1. Для заданного с клавиатуры значения N найти (2*N)!! по формуле:

(2*N)!! = 2*4*6*…*(2*N-2)*(2*N).

2. Вычислить предел последовательности $\{Y_n\}$ при $n\to\infty$, где $Y_n$ вычисляется по формуле

$$Y_n = \frac{Y_{n-2}+0.5\cdot Y_{n-3}}{Y_{n-2}^2+2Y_{n-3}^4+1.5}; n = 3,4,5\ldots$$

Значения $Y_0$, $Y_1$, $Y_2$ вводятся с клавиатуры. Вычисления прекратить при выполнении условия $|Y_n - Y_{n-1}|<\varepsilon$.

**10**

1. Для заданного с клавиатуры значения N найти (2*N+1)!! по формуле

(2*N+1)!! = 1*3*5*…*(2*N-1)*(2*N+1).

2. Последовательность функций $Y_n = Y_n(x)$, где $0\le x\le 1$ определяется следующим образом:

$$Y_1 = \frac{X}{2}; Y_n = \frac{1}{2}(X + Y_{n-1}^2); n = 2,3,4\ldots$$

При заданном x найти предел последовательности, принимая за таковой значение $Y_n$, удовлетворяющее условию $|Y_n - Y_{n-1}|<\varepsilon$.

**11**

1. Найти сумму всех целых чисел, кратных 5, из отрезка [A,B].

2. Последовательность функций $Y_n = Y_n(x)$, где $x > 0$ определяется следующим образом:

$Y_1 = x$; $Y_n = Y_{n-1}*(2 - x*Y_{n-1})$; n = 2,3,4…

При заданном X найти предел последовательности, принимая за таковой значение $Y_n$, удовлетворяющее условию $|Y_n - Y_{n-1}|<\varepsilon$.

**12**

1. Найти сумму всех целых чисел, кратных 7, из отрезка [A,B].
2. Найти предел произведения $P = \prod_{n=1}^{\infty}(1+\frac{1}{Y_n})$ для последовательности $\{Y_n\}$, пользуясь рекуррентной формулой

$$Y_1 = 1;\ Y_n = n*(Y_{n-1} + 1);\ n = 2,3,4\ldots$$

Вычисления закончить при выполнении условия $1/Y_n < \varepsilon$.

**13**

1. Найти сумму всех целых чисел, дающих при делении на 5 в остатке 3, из отрезка [A,B].
2. Вычислить $\sqrt[k]{A}$ - корень k-ой степени из положительного числа A, пользуясь последовательным приближением

$$X_0 = A; X_n = \frac{k-1}{k}X_{n-1} + \frac{A}{k \cdot X_{n-1}^{k-1}}; n = 1,2,3,...$$

За корень принять такое $X_n$, при котором $|X_n - X_{n-1}| < \varepsilon$.

**14**

1. Найти сумму всех целых чисел, дающих при делении на 7 в остатке 4, из отрезка [A,B].
2. Для приближенного решения уравнения Кеплера X-q*sin(X)=m, 0<q<1 полагают $X_0 = m$, $X_1 = m + q * \sin(X_0)$, …, $X_n = m + q * \sin(X_{n-1})$, …

При заданном m найти решение уравнения Кеплера, принимая за него такое $X_n$, при котором $|X_n - X_{n-1}| < \varepsilon$.

**15**

1. Пользуясь рекуррентной формулой, для заданного с клавиатуры m вычислить $S_m = \sum_{i=1}^{m}\sqrt{Y_i}$ если известны $Y_0$, $Y_1$, $Y_2$, а $Y_i$ вычисляется по формуле

$$Y_i = \lg\left|Y_{i-2}^2 + Y_{i-3} + 1\right|; i = 3,4,5,...$$

2. Вычислить предел последовательности $\{Y_n\}$ при $n \to \infty$, где $Y_n$ вычисляется по формуле: $Y_n = \frac{n}{\sqrt{n^2} + 1 + \sqrt{2*n^2 - 1}}$

Вычисления прекращаются при выполнении условия $|Y_n - Y_{n-1}| < \varepsilon$.

**16**

1. Пользуясь рекуррентной формулой, для заданного с клавиатуры m вычислить $Y_m$, если известны $Y_0$, $Y_1$, $Y_2$, а $Y_i$ вычисляется по формуле

$$Y_i = tg^2(Y_{i-1}) + Y_{i-2};\ i = 3,4,5,\ldots,m.$$

2. Найти предел последовательности $\lim_{n\to\infty}\frac{5*n}{3*\sqrt{(n^2+1)} + 2*\sqrt{(n^2-1)}}$ с точностью ε.

**17**

1. Пользуясь рекуррентной формулой, для заданного с клавиатуры m вычислить $S_m = \sum_{i=1}^{m} \ln(|Y_i| + 0.5)$, если известны $Y_0$, $Y_1$, $Y_2$, а $Y_i$ вычисляется по формуле

$$Y_i = Y_{i-1} + Y_{i-2}^2 - 2 * Y_{i-3}; i = 3,4,5,\ldots,\text{m}$$

2. Найти предел последовательности $\lim_{n\to\infty} \frac{n^3 + 5}{2 * n^3 + n^2 + 1}$ с точностью ε.

**18**

1. Пользуясь рекуррентной формулой, для заданного с клавиатуры m вычислить $Y_m$, если известны $Y_0$, $Y_1$, а $Y_i$ вычисляется по формуле

$$Y_i = \frac{2 * Y_{i-1} + Y_{i-2}}{3}; i = 2,3,4,\ldots,\text{m}$$

2. Найти сумму бесконечного ряда $\sum_{n=1}^{\infty} \frac{1}{n^2 * (\sin(n) + 1.1)}$ с точностью ε.

**19**

1. Пользуясь рекуррентной формулой, для заданного с клавиатуры m вычислить $Y_m$, если известны $Y_0$ ,$Y_1$,$Y_2$, а $Y_i$ вычисляется по формуле

$$Y_i = \sin^2(Y_{i-1}) + \cos^2(Y_{i-3}); i = 3,4,5, \ldots,m$$

2. Найти сумму бесконечного ряда $\sum_{n=1}^{\infty} \frac{1}{n * (n + A)}$ с точностью ε.

**20**

1. Пользуясь рекуррентной формулой, для заданного с клавиатуры m вычислить $S_m = \sum_{i=1}^{m} Y_i$, если известны $Y_0$, $Y_1$, $Y_2$, а $Y_i$ вычисляется по формуле

$$Y_i = \sin(Y_{i-1}) + \cos(Y_{i-3}); i = 3,4,5,\ldots,m.$$

2. Найти сумму бесконечного ряда $\sum_{n=1}^{\infty} \frac{1}{(5 * n - 1) * (5 * n + 1)}$ с точностью ε.

**21**

1. Пользуясь рекуррентной формулой, для заданного с клавиатуры m вычислить $S_m = \sum_{i=1}^{m} Y_i^2$ при известных $Y_0$ ,$Y_1$, если $Y_i$ вычисляется по формуле

$$Y_i = \sqrt{|\sin(Y_{i-1}) + \cos(Y_{i-2})|}; \ i=2,3,4,\ldots,m.$$

2. Найти сумму бесконечного ряда $\sum_{n=1}^{\infty} (-1)^n \frac{n}{2 * n^2 - 1}$ с точностью ε.

**22**

1. Члены последовательностей $\{X_i\}$ и $\{Y_i\}$ вычисляются по двум рекуррентным формулам. Вычислить $X_{20}$ ,$Y_{20}$, если

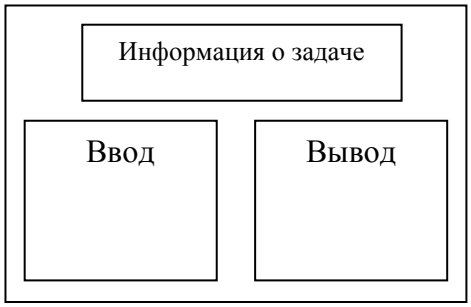
$$X_{i+1} = \sqrt{\frac{X_i * (Y_i + 5)^{-1}}{2}}; X_0 = 3.5;$$

$$Y_{i+1} = \sqrt{X_i + 1.6}; Y_0 = 2.2$$

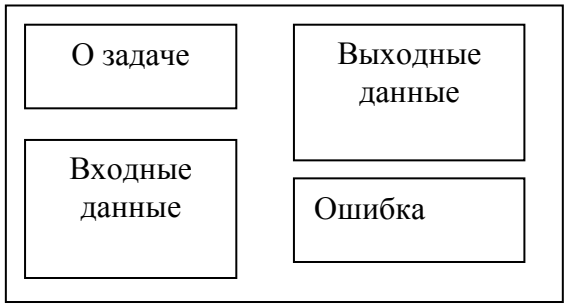
2. Найти сумму бесконечного ряда $\sum_{n=1}^{\infty} \frac{\cos(\frac{\pi}{n} + 30^0)}{n}$ с точностью ε.

## *Расположение окон*

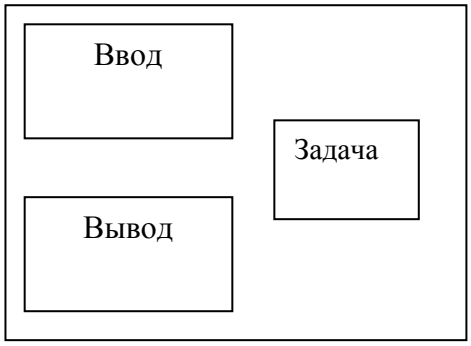
**1** *Расположение окон на экране:*



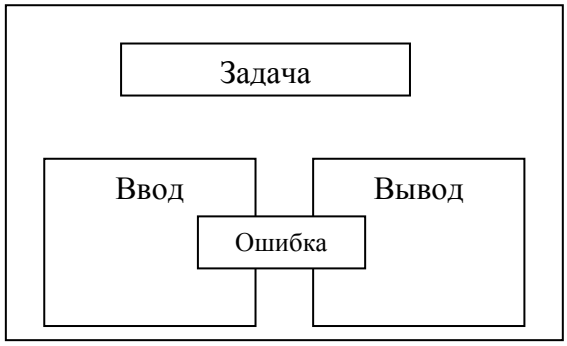

**2** Расположение окон на экране:



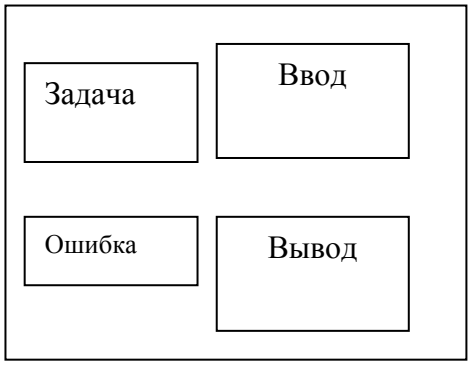

**3** Расположение окон на экране:



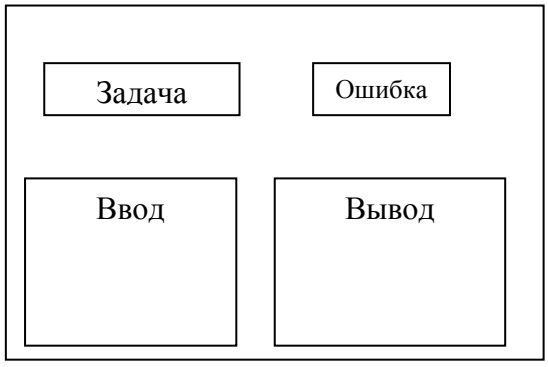

**4** Расположение окон на экране:



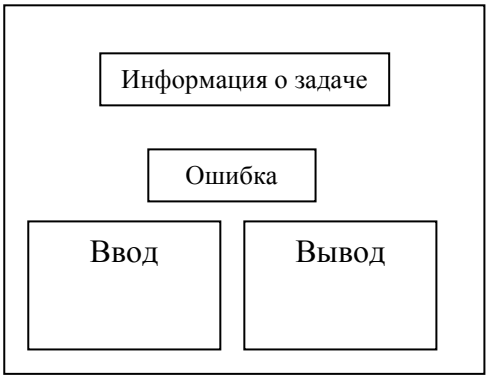

**5** Расположение окон на экране:



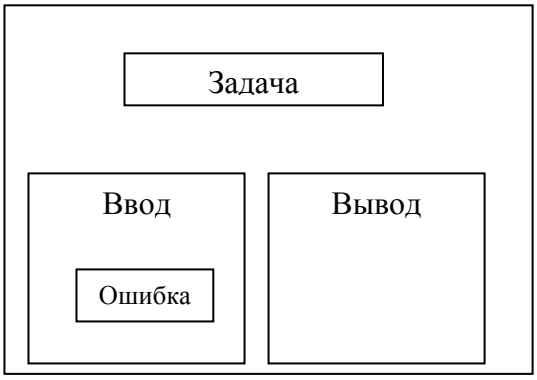

**6** Расположение окон на экране:



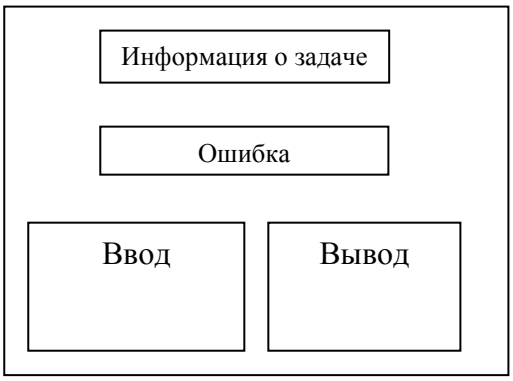
**7** Расположение окон на экране:



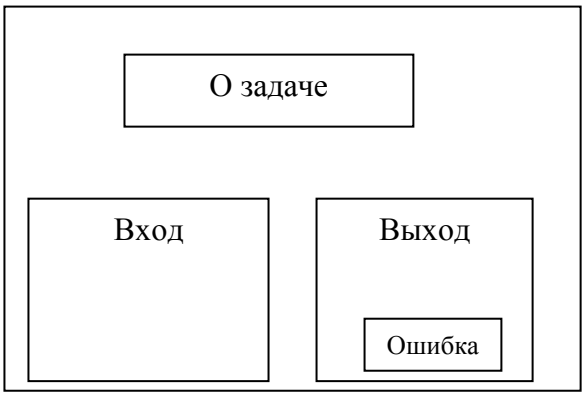

**8** Расположение окон на экране:



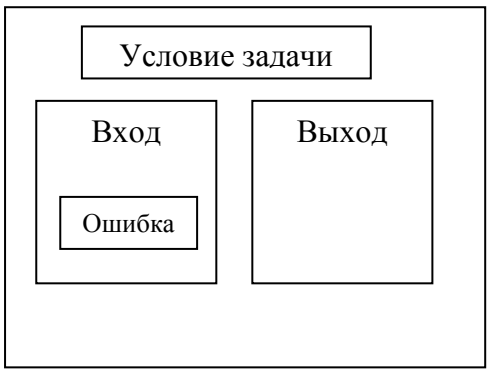

**9** Расположение окон на экране:



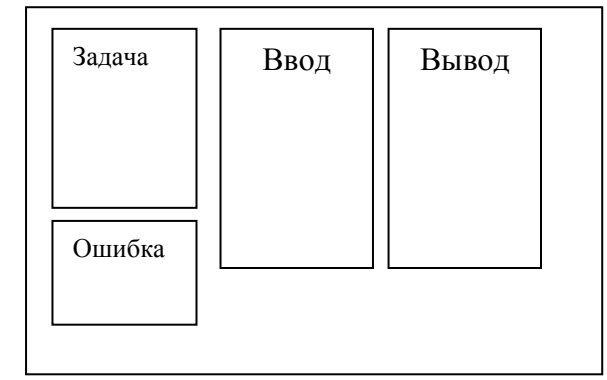

**10** Расположение окон на экране:



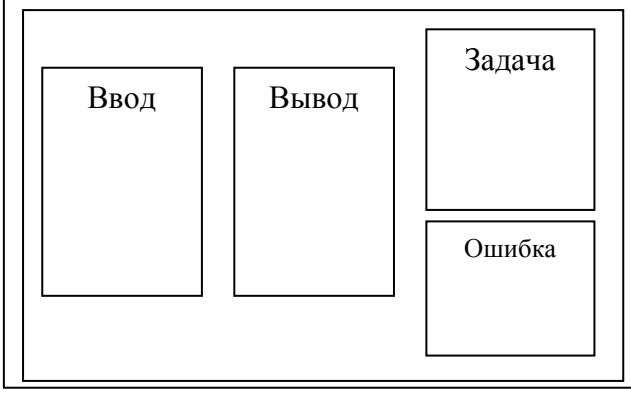

**11** Расположение окон на экране:



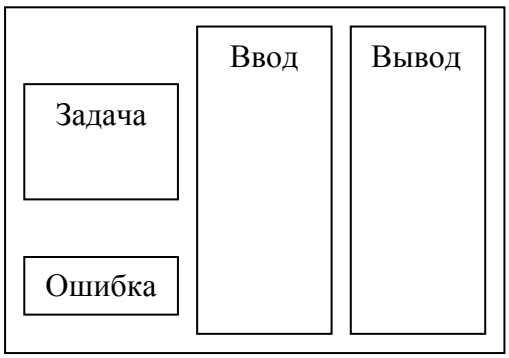

**12** Расположение окон на экране:



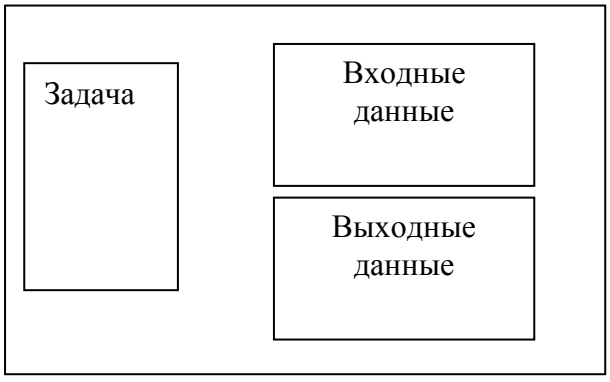
**13** Расположение окон на экране:



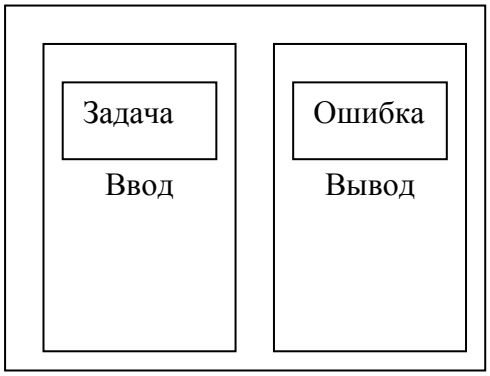

**14** Расположение окон на экране:



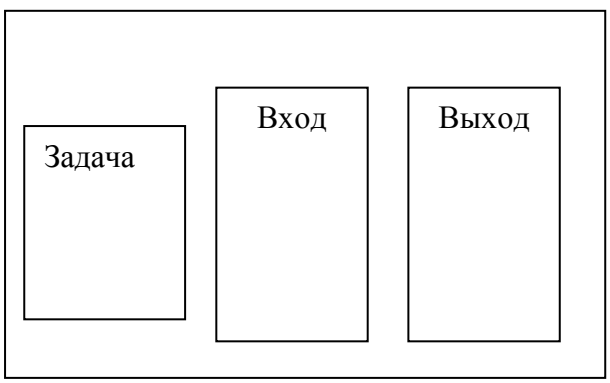

**15** Расположение окон на экране:



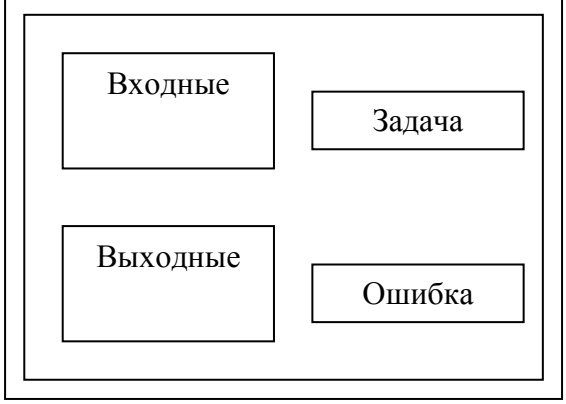

**16** Расположение окон на экране:




**17** Расположение окон на экране:




**18** Расположение окон на экране:



***Пример программы***

```
//1.Пользуясь рекуррентной формулой yi=yi-1 + yi-3^2, где i=3,4,...n,
//для заданного n вычислить yn, если известны y0, y1, y2.
//2.Последовательность {an} задана равенствами:
//  a1=0.5; an=n*(an-1+0.5).
//  Вычислить предел произведения (1+1/a1)*...*(1+1/an).
//  Вычисления закончить при (1/an) < eps.
#include<iostream.h>;
#include<math.h>;
#include<conio.h>;
#include<stdlib.h>
#include<limits.h>
int recur1(int n, int y0, int y1, int y2);
int recur2(int n, int y0, int y1, int y2);
int recur3(int n, int y0, int y1, int y2);
float predel1(float eps);
float predel2(float eps);
float predel3(float eps);
void okno(int x1, int y1, int x2, int y2, int colfona, int colbukv);

void main()
{int var, n, re1, re2, re3;
 float rez1, rez2, rez3, z, eps, y0, y1, y2;
 textbackground(BLACK);
 textcolor(15);
 clrscr();
 for(;;)
 {okno(20,1,60,6,1,15);
//Ввод исходных данных
   cprintf( "\n  Вид действия:\n\r");
   cprintf(" 1 - вычисление по рекуррентной формуле\n\r");
   cprintf(" 2 - вычисление предела произведения\n\r");
   cprintf(" 3 - завершение задачи\n\r");
   cprintf(" Введите вид действия ->");
   cin >> var;
   switch(var)
     {case 1: okno(1, 10, 37, 15, 2, 15);
//Ввод исходных данных для первой задачи
           cprintf(" Введите n ->");
           cin >> n;
           cprintf(" Введите y0, y1, y2 ->");
           cin >> y0 >> y1 >> y2;
           re1 = recur1(n, y0, y1, y2);
           re2 = recur2(n, y0, y1, y2);
           re3 = recur3(n, y0, y1, y2);
           okno(40,10,80,15,4,15);
//Вывод результата
           cprintf(" Для цикла WHILE результат    =%d\n\r",re1);
           cprintf(" Для цикла DO..WHILE результат=%d\n\r",re2);
           cprintf(" Для цикла FOR результат      =%d\n\r",re3);
           break;
```

```
      case 2: okno(1, 10, 37, 15, 2, 15);
//Ввод исходных данных для второй задачи
           cprintf(" Введите точность вычисления\n\r");
           cin >> eps;
           rez1 = predel1(eps);
           rez2 = predel2(eps);
           rez3 = predel3(eps);
           okno(40, 10, 80, 15, 4, 15);
//Вывод результата
           cprintf(" Для цикла WHILE результат    =%f\n\r",rez1);
           cprintf(" Для цикла DO..WHILE результат=%f\n\r",rez2);
           cprintf(" Для цикла FOR результат      =%f\n\r",rez3);
           break;
      default: abort();
     }//switch
 }//for
}

//Вывод окна на экран
void okno(int x1, int y1, int x2, int y2, int colfona, int colbukv)
{window(x1, y1, x2, y2);
 textbackground(colfona);
 textcolor(colbukv);
 clrscr();
}

//вычисление значения рекуррентного выражения циклом while
int recur1(int n, int y0, int y1, int y2)
{int i = 3, y;
 while(i <= n)
   {y = y2 + y0 * y0;
    y0 = y1;
    y1 = y2;
    y2 = y;
    i++;
   }
 return(y);
}

//вычисление значения рекуррентного выражения циклом do..while
int recur2(int n, int y0, int y1, int y2)
{int i = 3, y;
 do
  {y = y2 + y0 * y0;
   y0 = y1;
   y1 = y2;
   y2 = y;
   i++;
  }
 while(i <= n);
 return(y);
}
```

```
//вычисление значения рекуррентного выражения циклом for
int recur3(int n, int y0, int y1, int y2)
{int i, y;
 for(i = 3; i <= n; i++)
    {y = y2 + y0 * y0;
     y0 = y1;
     y1 = y2;
     y2 = y;
    }
 return(y);
}

//вычисление предела произведения циклом while
float predel1(float eps)
{float pr = 1, an = 0.5;
 int n = 1;
 while(fabs(1 / an) > eps)
   {pr *= (0.5 + 1 / an);
    n++;
    an = n * (an + 1);
   }
 return(pr);
}

//вычисление предела произведения циклом do..while
float predel2(float eps)
{float an = 0.5, pr = 1;
 int n = 1;
 do
  {pr *= (0.5 + 1 / an);
   n++;
   an = n * (an + 1);
  }
 while (fabs(1 / an) > eps);
 return(pr);
}

//вычисление предела произведения циклом for
float predel3(float eps)
{float an = 0.5, pr = 0.5 + 1 / an;
 int n;
 for(n = 2; n < INT_MAX; n++)
    {an = n * (an + 1);
     if(fabs(1 / an) > eps) pr *= (0.5 + 1 / an);
     else break;
    }
 return(pr);
}
```

## Лабораторная работа №5
## Суммирование рядов

***Цель лабораторной работы:*** *применение технологии структурного программирования для решения задач суммирования рядов.*

***Задание на программирование:*** *используя технологию структурного программирования, разработать программу вычисления суммы ряда с заданной точностью в заданном интервале допустимых значений аргумента.*

*Программа должна формировать таблицу, содержащую значения аргумента ряда, суммы ряда, количество слагаемых и контрольные значения суммы, полученные с помощью стандартных функций библиотеки.*

***Порядок выполнения работы:***

1) Получить у преподавателя индивидуальное задание и выполнить *постановку задачи*: сформулировать условие, определить входные и их ограничения, определить вид выходной таблицы значений.

2) Разработать *математическую модель*:

- вывести *рекуррентную формулу* для расчета очередного слагаемого;
- описать начальные установки номера слагаемого, слагаемого, суммы;
- описать процесс накопления суммы.

3) Построить *схему алгоритма*. Обосновать выбор циклических управляющих структур.

4) Составить программу на языке C/C++.

5) Использовать *оконный интерфейс* предыдущей лабораторной работы.

*Входные данные* вводить с клавиатуры по запросу.

*Выходные данные* выводить на экран в форме таблицы с графами:
аргумент, сумма, количество слагаемых, контрольное значение суммы.

6) Проверить и продемонстрировать преподавателю работу программы, при этом значение суммы должно совпадать с соответствующим контрольным значением (с заданной точностью). Выходная таблица должна содержать от 5 до 10 строк.

7) Оформить *отчет о лабораторной работе* в составе: *постановка задачи, математическая модель, схема алгоритма решения, текст программы, контрольные примеры.*

## *Варианты индивидуальных заданий*

**1** ***arctg x*** = $\sum_{n=0}^{\infty}(-1)^n \cdot \frac{x^{2\cdot n+1}}{2\cdot n+1} = x - \frac{x^3}{3} + \frac{x^5}{5} - \frac{x^7}{7} + \ldots$ , $|X| < 1$.

**2** ***arctg x*** = $\frac{\pi}{2} - \sum_{n=0}^{\infty}(-1)^{n+1} \cdot \frac{1}{(2\cdot n+1)\cdot x^{2n+1}} = \frac{\pi}{2} - (\frac{1}{x} - \frac{1}{3\cdot x^3} + \frac{1}{5\cdot x^5} - \frac{1}{7\cdot x^7} + \ldots)$ , $X > 1$.

**3** ***arcth x*** = $\sum_{n=0}^{\infty} \frac{x^{2\cdot n+1}}{2\cdot n+1} = x + \frac{x^3}{3} + \frac{x^5}{5} + \frac{x^7}{7} + \ldots$, $|X| < 1$.

**4** ***arcth x*** = $\sum_{n=0}^{\infty} \frac{1}{(2\cdot n+1)\cdot x^{2\cdot n+1}} = \frac{1}{x} + \frac{1}{3\cdot x^3} + \frac{1}{5\cdot x^5} + \frac{1}{7\cdot x^7} + \ldots$ , $|X| > 1$.

**5** ***ln x*** = $\sum_{n=1}^{\infty}(-1)^{n+1} \cdot \frac{(x-1)^n}{n} = \frac{(x-1)^1}{1} - \frac{(x-1)^2}{2} + \frac{(x-1)^3}{3} - \frac{(x-1)^4}{4} + \ldots$ , $0 < X < 2$.

**6** ***ln (1+x)*** = $\sum_{n=1}^{\infty}(-1)^{n+1} \cdot \frac{x^n}{n} = x - \frac{x^2}{2} + \frac{x^3}{3} - \frac{x^4}{4} + \ldots$ , $-1 < X < 1$.

**7** ***ln (1-x)*** = $-\sum_{n=1}^{\infty} \frac{x^n}{n} = -(x + \frac{x^2}{2} + \frac{x^3}{3} + \frac{x^4}{4} + \ldots)$, $X < 1$.

**8** ***ln*** $\left(\frac{1+x}{1-x}\right) = 2\cdot\sum_{n=0}^{\infty}\frac{x^{2\cdot n+1}}{2\cdot n+1} = 2\cdot\left(x + \frac{x^3}{3} + \frac{x^5}{5} + \frac{x^7}{7} + \ldots\right)$ , $|X| < 1$.

**9** ***ln*** $\left(\frac{x+1}{x-1}\right) = 2\cdot\sum_{n=0}^{\infty}\frac{1}{(2\cdot n+1)\cdot x^{2\cdot n+1}} = 2\cdot\left(\frac{1}{x} + \frac{1}{3\cdot x^3} + \frac{1}{5\cdot x^5} + \frac{1}{7\cdot x^7} + \ldots\right)$ , $|X| > 1$.

**10** $e^x \cdot (1+x) = \sum_{n=0}^{\infty}\frac{x^n\cdot(n+1)}{n!} = 1 + \frac{2\cdot x}{1!} + \frac{3\cdot x^2}{2!} + \frac{4\cdot x^3}{3!} + \ldots$ , $|X| < 2.4$ .

**11** $e^{-x^2} = \sum_{n=0}^{\infty}(-1)^n \cdot \frac{x^{2\cdot n}}{n!} = \frac{x^0}{0!} - \frac{x^2}{1!} + \frac{x^4}{2!} - \frac{x^6}{3!} + \ldots$ , $X < 1$.

**12** $\ln x = 2 \cdot \sum_{n=0}^{\infty} \frac{(x-1)^{2 \cdot n+1}}{(2 \cdot n+1) \cdot (x+1)^{2 \cdot n+1}} = 2 \cdot \left( \frac{x-1}{x+1} + \frac{(x-1)^3}{3 \cdot (x+1)^3} + \frac{(x-1)^5}{5 \cdot (x+1)^5} + \ldots \right)$ , $X > 0$.

**13** $\ln x = \sum_{n=1}^{\infty} \frac{(x-1)^n}{n \cdot x^n} = \frac{x-1}{x} + \frac{(x-1)^2}{2 \cdot x^2} + \frac{(x-1)^3}{3 \cdot x^3} + \ldots$ , $X > 0.5$.

**14** $\sin x = \sum_{n=1}^{\infty} (-1)^{n-1} \cdot \frac{x^{2 \cdot n-1}}{(2 \cdot n-1)!} = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \ldots$ , $|X| < \infty$.

**15** $\cos x = \sum_{n=0}^{\infty} (-1)^n \cdot \frac{x^{2 \cdot n}}{(2 \cdot n)!} = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6!} + \frac{x^8}{8!} - \ldots$ , $|X| < \infty$.

**16** $\operatorname{sh} x = \sum_{n=1}^{\infty} \frac{x^{2 \cdot n-1}}{(2 \cdot n-1)!} = x + \frac{x^3}{3!} + \frac{x^5}{5!} + \frac{x^7}{7!} + \ldots$ , $|X| < \infty$, $\operatorname{sh} x = \frac{e^x - e^{-x}}{2}$.

**17** $\operatorname{ch} x = \sum_{n=0}^{\infty} \frac{x^{2 \cdot n}}{(2 \cdot n)!} = 1 + \frac{x^2}{2!} + \frac{x^4}{4!} + \frac{x^6}{6!} + \frac{x^8}{8!} + \ldots$ , $|X| < \infty$, $\operatorname{ch} x = \frac{e^x + e^{-x}}{2}$.

**18** $\sin^2 x = \sum_{n=1}^{\infty} (-1)^{n+1} \cdot \frac{2^{2 \cdot n-1} \cdot x^{2 \bullet n}}{(2 \cdot n)!} = \frac{2^1 \cdot x^2}{2!} - \frac{2^3 \cdot x^4}{4!} + \frac{2^5 \cdot x^6}{6!} - \frac{2^7 \cdot x^8}{8!} + \ldots$ , $X < 1$.

**19** $\cos^2 x = 1 - \sum_{n=1}^{\infty} (-1)^{n+1} \cdot \frac{2^{2 \cdot n-1} \cdot x^{2 \cdot n}}{(2 \cdot n)!} = 1 - \left( \frac{2^1 \cdot x^2}{2!} - \frac{2^3 \cdot x^4}{4!} + \frac{2^5 \cdot x^6}{6!} - \frac{2^7 \cdot x^8}{8!} + \ldots \right)$ , $X < 1$.

**20** $\operatorname{arcctg} x = \frac{\pi}{2} + \sum_{n=0}^{\infty} (-1)^{n+1} \cdot \frac{x^{2 \cdot n+1}}{2 \cdot n+1} = \frac{\pi}{2} - \left( x - \frac{x^3}{3} + \frac{x^5}{5} - \frac{x^7}{7} + \ldots \right)$ , $|X| < 1$.

**21** $\operatorname{arctg} x = \frac{\pi}{2} + \sum_{n=0}^{\infty} (-1)^{n+1} \cdot \frac{1}{(2 \cdot n+1) \cdot x^{2 \cdot n+1}} = \frac{\pi}{2} - \left( \frac{1}{x} - \frac{1}{3 \cdot x^3} + \frac{1}{5 \cdot x^5} - \frac{1}{7 \cdot x^7} + \ldots \right)$ , $|X| > 1$.

**22** $\operatorname{arcctg} x = \sum_{n=0}^{\infty} (-1)^n \cdot \frac{1}{(2 \cdot n+1) \cdot x^{2 \cdot n+1}} = \left( \frac{1}{x} - \frac{1}{3 \cdot x^3} + \frac{1}{5 \cdot x^5} - \frac{1}{7 \cdot x^7} + \ldots \right)$ , $|X| > 1$.

**23** $\operatorname{arcsin} x = x + \sum_{n=1}^{\infty} \frac{1 \cdot 3 \cdot 5 \cdot \ldots \cdot (2 \cdot n-1) \cdot x^{2 \cdot n+1}}{2 \cdot 4 \cdot 6 \cdot \ldots \cdot (2 \cdot n) \cdot (2 \cdot n+1)} = x + \frac{x^3}{2 \cdot 3} + \frac{1 \cdot 3 \cdot x^5}{2 \cdot 4 \cdot 5} + \frac{1 \cdot 3 \cdot 5 \cdot x^7}{2 \cdot 4 \cdot 6 \cdot 7} + \ldots$ , $|X| < 1$.

**24** ***arccos x*** $= \frac{\pi}{2} - x - \sum_{n=1}^{\infty} \frac{1\cdot 3\cdot 5\cdot \ldots \cdot (2\cdot n-1)\cdot x^{2\cdot n+1}}{2\cdot 4\cdot 6\cdot \ldots \cdot (2\cdot n)\cdot (2\cdot n+1)} = \frac{\pi}{2} - (x + \frac{x^3}{2\cdot 3} + \frac{1\cdot 3\cdot x^5}{2\cdot 4\cdot 5} + \frac{1\cdot 3\cdot 5\cdot x^7}{2\cdot 4\cdot 6\cdot 7} + \ldots)$ , $|X| < 1$.

**25** ***arcsh x*** $= x + \sum_{n=1}^{\infty} (-1)^n \cdot \frac{1\cdot 3\cdot 5\cdot \ldots \cdot (2\cdot n-1)\cdot x^{2\cdot n+1}}{2\cdot 4\cdot 6\cdot \ldots \cdot (2\cdot n)\cdot (2\cdot n+1)} = x - \frac{x^3}{2\cdot 3} + \frac{1\cdot 3\cdot x^5}{2\cdot 4\cdot 5} - \frac{1\cdot 3\cdot 5\cdot x^7}{2\cdot 4\cdot 6\cdot 7} + \ldots$ , $|X| < 1$.

**26** ***arcch x*** $= \ln(2x) - \sum_{n=1}^{\infty} \frac{1\cdot 3\cdot 5\cdot \ldots \cdot (2\cdot n-1)}{2\cdot 4\cdot 6\cdot \ldots \cdot (2\cdot n)\cdot (2\cdot n)\cdot x^{2n}} = \ln(2x) - \frac{1}{2\cdot 2\cdot x^2} - \frac{1\cdot 3}{2\cdot 4\cdot 4\cdot x^4} - \frac{1\cdot 3\cdot 5}{2\cdot 4\cdot 6\cdot 6\cdot x^6} - \ldots$ , $X > 1$.

**27** $\sin^3 x = \frac{1}{4}\cdot \sum_{n=1}^{\infty} (-1)^{n+1} \cdot \frac{3^{2\cdot n+1}-3}{(2\cdot n+1)!}\cdot x^{2\cdot n+1} = \frac{1}{4}\cdot \left( \frac{3^3-3}{3!}\cdot x^3 - \frac{3^5-3}{5!}\cdot x^5 + \frac{3^7-3}{7!}\cdot x^7 - \ldots \right)$ , $X < 1$.

**28** $\cos^3 x = \frac{1}{4}\cdot \sum_{n=0}^{\infty} (-1)^n \cdot \frac{3^{2\cdot n}+3}{(2\cdot n)!}\cdot x^{2\cdot n} = \frac{1}{4}\cdot \left( \frac{3^0+3}{0!}\cdot x^0 - \frac{3^2+3}{2!}\cdot x^2 + \frac{3^4+3}{4!}\cdot x^4 - \ldots \right)$ , $X < 1$.

## Проверочные формулы

$$\arcsin x = \operatorname{arctg} \frac{x}{\sqrt{1-x^2}}$$

$$\arccos x = \frac{\pi}{2} - \operatorname{arctg} \frac{x}{\sqrt{1-x^2}}$$

$$\operatorname{arcctg} x = \frac{\pi}{2} - \operatorname{arctg} x$$

$$\operatorname{arcth} x = \frac{1}{2}\cdot \ln \frac{x+1}{x-1}, x > 1$$

$$\operatorname{arcsh} x = \ln\left(x + \sqrt{x^2+1}\right)$$

### *Пример программы*

```
//Вычислить π/4=1 - 1/3 + 1/5 - 1/7 +..для различных значений точности.
//Результаты представить в виде таблицы:
//точность, сумма, количество слагаемых, контрольное значение.

#include<stdio.h>
#include<math.h>
#include<conio.h>
#include<limits.h>

void windo(int x1,int y1,int x2,int y2,int colf,int colb);
void main()
{int vid, n;
 float eps, epsn, epsk, h, pr, rez;
 textbackground(BLACK) ;
 clrscr() ;
 for(;;)
 {windo(20,1,55,6,3,15);
  gotoxy((55 - 20 - 13) / 2,1);
//Ввод исходных данных
  cprintf("Вид действия:\n\r");
  cprintf("\r\n   1 - получение таблицы значений\n\r");
  cprintf("   2 - завершение программы\n\r");
  cprintf("   Выберите вид действия ->");
  scanf("%d",&vid);
  if (vid == 1)
  {window(1,1,80,25);
   textbackground(BLACK);
   clrscr();
   windo(20,1,55,6,3,15);
   gotoxy((55 - 20 - 13) / 2,1);
   cprintf("Вид действия:\n\r");
   cprintf("\r\n   1 - получение таблицы значений\n\r");
   cprintf("   2 - завершение программы\n\r");
   cprintf("   Выберите вид действия ->");
   windo(20,8,55,12,2,15);
   gotoxy((55 - 20 - 21) / 2,1);
   cprintf("Ввод исходных данных:");
//Ввод исходных данных
   cprintf("\r\n Введите нач знач точн  ");
// \r для возврата в начало строки (в случае наличия окон)
   scanf("%f", &epsn);
   if((epsn <= 0) || (epsn > 0.1))
     {windo(10,13,45,15,4,15);
      cprintf("\n Ошибка! Значение д.б. >0 и <0.1");
      getchar();getchar();
      return;
     }
   cprintf("\r Введите кон знач точн  ");
   scanf("%f", &epsk);
   if((epsk <= 0) || (epsk > 0.1))
```

```
      {windo(10,13,45,15,4,15);
       cprintf("\n Ошибка! Значение д.б. >0 и <0.1");
       getchar();getchar();
       return;
      }
   cprintf("\r Введите шаг измен точн ");
   scanf("%f", &h);
   if(h <= 0)
     {windo(10,13,45,15,4,15);
      cprintf("\n Ошибка! Значение д.б. >0");
      getchar();getchar();
      return;
     }
//Вывод заголовка таблицы
   windo(10,13,65,25,4,15);
   gotoxy((65 - 10 - 10) / 2,1);
   cprintf("Результат:");
   cprintf("\r\n  Точность|    Сумма   |Кол.слаг.|Контр значен\n\r");
//Вычисление суммы
   eps =epsn;
   do{n = 0;
      rez = 0;
      pr = 1;
      while (fabs(pr) > eps)
       {rez += pr;
        n++;
        pr *= - (2 * n - 1.) / (2 * n + 1);
        if(n >= INT_MAX)
          {cprintf("\r Точность не достигнута!!");
           getchar();getchar();
           return;
          }
       }
      cprintf(" %9.6f%12.8f%8i%15.8f\n\r",eps,rez,n,M_PI / 4);
      eps += h;
     }while(eps <= epsk);
  }
  else break;
 }
}

//Вывод окна на экран
void windo(int x1,int y1,int x2,int y2,int colf,int colb)
{window(x1, y1, x2, y2);
 textbackground(colf);
 textcolor(colb);
 clrscr();
}
```

## Лабораторная работа №6
## Обработка массивов

***Цель лабораторной работы:*** *изучение структурной организации массивов и способов доступа к их элементам; совершенствование навыков структурного программирования на языке С/С++ при решении задач обработки массивов.*

***Задание на программирование:*** *используя технологию структурного программирования, разработать программу обработки одномерных и двумерных (матриц) массивов в соответствии с индивидуальным заданием.*

***Порядок выполнения работы:***

1) Получить у преподавателя индивидуальное задание и выполнить *постановку задачи*: сформулировать условие, определить входные и выходные данные, их ограничения.

2) Разработать *математическую модель*: описать с помощью формул и рисунков структуру массивов и процесс их преобразования.

3) Построить *схему алгоритма* решения задачи.

4) Составить программу на языке С/С++.

5) Использовать *оконный интерфейс* предыдущей лабораторной работы.
*Входные данные* вводить с клавиатуры по запросу.
*Выходные данные* выводить на экран с пояснениями.

6) Проверить и продемонстрировать преподавателю работу программы на *полном* наборе тестов, в том числе с ошибочными входными данными. Входные и выходные массивы должны выводиться в одном и том же формате.

7) Оформить *отчет о лабораторной работе* в составе: *постановка задачи, математическая модель, схема алгоритма решения, текст программы, контрольные примеры.*

## *Варианты индивидуальных заданий*

**1**

1) Дан массив $b_1,b_2,\ldots,b_{2n}$. Написать программу построения массивов $x_1,x_2,\ldots,x_n$ и $y_1,y_2,\ldots,y_n$, элементы которых равны соответственно значениям: $b_1,b_3,\ldots,b_{2n-1}$ и $b_2,b_4,\ldots,b_{2n}$.

2) В заданной матрице поменять местами первую строку и строку, содержащую наибольший элемент матрицы.

**2**

1) Дан целочисленный массив $a_1,a_2,\ldots,a_m$. Из абсолютных величин его элементов выбрать наибольшую. Далее построить массив, i-й элемент которого равен нулю, если $|a_i|$ не совпадает с выбранным значением, и равен 1 в противном случае.

2) В заданной матрице поменять местами последний столбец и столбец, содержащий наименьший элемент матрицы.

**3**

1) Написать программу построения массива с элементами:
$a_1$, $a_1+a_2$, $a_1+a_2+a_3$, $a_1+a_2+a_3+\ldots+a_n$
по данному массиву $a_1$, $a_2,\ldots$, $a_n$.

2) В заданной матрице поменять местами две строки: строку, содержащую максимальный элемент матрицы, и строку, содержащую минимальный элемент матрицы.

**4**

1) В вещественном массиве $x_1,x_2,\ldots,x_n$ заменить нулем все отрицательные элементы, предшествующие его максимальному элементу.

2) В заданной матрице поменять местами главную и побочную диагонали.

**5**

1) Даны массивы $a_1$, $a_2$ …, $a_n$ и $b_1$, $b_2,\ldots$, $b_n$. Получить массив C, элементы которого:
$a_1$, $b_1$, $a_2$, $b_2$, … $a_n$, $b_n$.

2) В заданной матрице поменять местами первый столбец со столбцом, содержащим наибольший элемент матрицы.

**6**

1) Дан вещественный массив $x_1,x_2,\ldots,x_m$. Все его элементы, следующие за наибольшим элементом, заменить значением b.

2) В заданной матрице поменять местами среднюю строку и средний столбец.

**7**

1) Даны вещественные массивы $x_1,x_2,\ldots,x_n$ и $y_1,y_2,\ldots,y_n$. Преобразовать их по правилу: большее из значений $x_i$ и $y_i$ принять в качестве нового значения $x_i$, а меньшее – в качестве нового значения $y_i$.

2) В заданной матрице поменять местами последнюю строку со строкой, содержащей наибольший элемент матрицы.

**8**

1) Дан целочисленный массив $a_1,a_2,\ldots,a_n$. Если в массиве нет ни одной компоненты с заданным значением К, то первую по порядку компоненту этого массива, большую всех остальных компонент, заменить значением К.

2) В заданной матрице поменять местами первую строку и первый столбец.

**9**

1) Написать программу, осуществляющую циклический сдвиг компонент массива $x_1,x_2,\ldots,x_n$ (n>=2) на одну позицию влево, то есть получающую массив $x_2,x_3,\ldots,x_n,x_1$.

2) В заданной матрице поменять местами последний столбец со столбцом, содержащим наибольший элемент матрицы.

**10**

1) Дан вещественный массив $a_1,a_2,\ldots,a_n$. Если в этом массиве есть хотя бы один элемент, значение которого меньше P, то все отрицательные элементы массива заменить их квадратами, в противном случае массив a умножить на число b.

2) В заданной матрице поменять местами последнюю строку со строкой, содержащей наименьший элемент матрицы.

**11**

1) Написать программу вычисления величины K, обратной произведению тех элементов массива $b_1,b_2,\ldots,b_n$, для которых выполнимо: $2^i<b_i<i!$. Если таких элементов нет, то ответом должно служить сообщение.

2) В заданной матрице поменять местами первый столбец со столбцом, содержащим наибольший элемент главной диагонали.

**12**

1) Преобразовать массив $a_1,a_2,\ldots,a_n$ так, чтобы его элементы расположились в обратном порядке: $a_n,a_{n-1},\ldots,a_1$.

2) В заданной матрице поменять местами две строки: строку с указанным номером и строку, содержащую наименьший элемент матрицы.

**13**

1) Написать программу выбора среди элементов массива $a_1,a_2,\ldots,a_n$ наибольшего среди остающихся после выбрасывания наибольшего и всех ему равных. Предполагается, что не все элементы равны между собой.

2) В заданной матрице поменять местами последний столбец и побочную диагональ.

**14**

1) Из массива $a_1,a_2,\ldots,a_{3n}$ получить массив $b_1,b_2,\ldots,b_n$, очередная компонента которого равна среднему арифметическому тройки очередных компонент массива a.

2) В заданной матрице поменять местами два столбца: столбец, содержащий максимальный элемент матрицы, и столбец, содержащий минимальный элемент матрицы.

**15**

1) Дан целочисленный массив $b_1,b_2,\ldots,b_n$. Если элементы этого массива не образуют убывающей последовательности, то заменить его отрицательные элементы единицами.

2) В заданной матрице поменять местами первую строку и строку, содержащую максимальный элемент матрицы.

**16**

1) Дан целочисленный массив $a_1,a_2,\ldots,a_n$, среди элементов которого могут быть равные. Из каждой группы равных между собой элементов нужно оставить только один, выбросив все остальные. Освободившийся хвост массива заполнить нулями.

2) В заданной матрице поменять местами первый столбец и побочную диагональ.

**17**

1) Дан вещественный массив $a_1,a_2,\ldots,a_n$. Если в этом массиве есть хотя бы один элемент, принадлежащий отрезку [x,y], то все элементы, не принадлежащие этому отрезку, заменить значением K.

2) В заданной матрице поменять местами последнюю строку со строкой, содержащей минимальный элемент матрицы.

**18**

1) Дан массив $a_1,a_2,,\ldots,a_n$. Переставить его элементы так, чтобы в начале массива расположились все его неотрицательные элементы, а в конце – отрицательные.

2) В заданной матрице поменять местами последний столбец и столбец, содержащий минимальный элемент матрицы.

**19**

1) Написать программу выполнения следующего задания: из всех непрерывных участков массива $a_1,a_2,\ldots,a_n$, состоящих из нулей, выбрать наибольший по длине. Вывести индексы его начала и конца.

2) В заданной матрице поменять местами последнюю строку со строкой, содержащей максимальный элемент матрицы.

**20**

1) Написать программу, осуществляющую циклический сдвиг компонент массива $x_1,x_2,\ldots,x_n$ (n>=2) на одну позицию вправо, т.е. получающую массив $x_n,x_1,x_2,\ldots,x_{n-1}$.

2) В заданной матрице поменять местами последний столбец со столбцом, содержащим максимальный элемент матрицы.

**21**

1) Дан вещественный массив $x_1,x_2,\ldots,x_m$. Все его элементы, предшествующие наибольшему элементу, заменить значением c.

2) В заданной матрице поменять местами первую строку и главную диагональ.

**22**

1) Дан вещественный массив $x_1,x_2,\ldots,x_m$. Все его положительные элементы, следующие за наименьшим элементом, заменить значением d.

2) В заданной матрице поменять местами главную диагональ и последний столбец.

### *Пример программы на обработку одномерного массива*

```
//Найти и вывести номер элемента введенного с клавиатуры массива целых чисел,
//для которого сумма разностей с соседними элементами максимальна.
//Для крайних элементов использовать циклическое замыкание.
#include<stdio.h>
#include<conio.h>
#include<string.h>
#include<math.h>
const int RAZ = 10;   //размер массива
int nomer(int a[], int &max);
void inputmas(int a[]);
void okno(int x1,int y1,int x2,int y2,int bkcol,int colb,char zag[15]);
void main()
{int a[RAZ];               //массив
 int nom;                  //номер искомого элемента
 int max;                  //значение максимальной разности
 okno(1,1,80,25,BLACK,WHITE,"");
 okno(15,1,65,5,WHITE,BLUE,"Описание");
 cprintf("\r\n В массиве целых чисел найти номер");
 cprintf("\n\r элемента, для которого сумма разностей");
 cprintf("\n\r с соседними элементами максимальна");
 okno(15,15,65,20,RED,WHITE,"Результат поиска");
 okno(15,7,65,13,WHITE,BLUE,"Окно ввода");
//Ввод исходных данных
 inputmas(a);
//Поиск номера элемента
 nom = nomer(a, max);
 okno(15,15,65,20,RED,WHITE,"Результат поиска");
 cprintf("\n\r Искомый номер элемента массива: %i", nom);
 cprintf("\n\r Значение элемента: %i, сумма разностей= %i", a[nom], max);
 cprintf("\n\r Для завершения нажмите <Enter>");
 getch();
}

int nomer(int a[], int &max)
{int pr;              //текущее значение разности
 int imax = 0;        //за максимум принимаем первый по счету элемент
 max = abs(a[RAZ - 1] - a[0]) + abs(a[1] - a[0]);
 for(int i = 1; i < RAZ - 1; i++)
    if(max < (pr = abs(a[i-1] - a[i]) + abs(a[i+1] - a[i])))
      {imax = i;
       max  = pr;
      }
 if(max < abs(a[0] - a[RAZ - 1]) + abs(a[RAZ - 2] - a[RAZ-1]))
   imax = RAZ - 1;
 return imax;
}

void okno(int x1,int y1,int x2,int y2,int bkcol,int colb,char zag[15])
{window(x1,y1,x2,y2);
 textbackground(bkcol);
 textcolor(colb);
 clrscr();
 gotoxy((x2 - x1 - strlen(zag)) / 2,1);
 cprintf("%s\n\r",zag);
}
```

```
void inputmas(int a[])
{cprintf(" Введите в одной строке элементы массива,\n\r");
 cprintf(" состоящего из %i целых чисел, и нажмите <Enter>\n\r", RAZ);
 cprintf(" ->");
 for(int i = 0; i < RAZ; i++)
    scanf("%i", &a[i]);
}
```

## *Пример программы на обработку двумерного массива (матрицы)*

```
//Программа находит строку введенного с клавиатуры двумерного массива целых
//чисел, содержащую максимальную сумму элементов

#include <stdio.h>
#include <conio.h>
#include<stdlib.h>
#include<string.h>
const RAZ = 10;       //размер одного измерения массива

void inputmatr(int matr[][RAZ],int &m, int &n);
void okno(int x1,int y1,int x2,int y2,int bkcol,int colb,char zag[15]);
int exist(int matr[][RAZ],int n,int x,int p,int k);
void poisk_st(int m,int n,int matr[][RAZ],int &max,int &jmax);
void outmatr(int m,int n,int matr[][RAZ],int imax);

void main()
{int a[RAZ][RAZ];     //массив
 int imax;            //номер строки с максимальной суммой элементов
 int max;             //максимальная сумма элементов
 int m;               //число строк
 int n;               //число столбцов

 okno(1,1,80,25,BLACK,WHITE,"");
 okno(15,1,60,4,WHITE,BLUE,"Описание");
 cprintf("\r\n В матрице целых чисел найти номер строки,");
 cprintf("\n\r содержащей максимальную сумму элементов");
 okno(10,10,65,25,RED,WHITE,"Результат");
 okno(15,6,60,8,WHITE,BLUE,"Окно ввода");
//ввод исходных данных
  inputmatr(a,m,n);

//поиск строки с максимальной суммой элементов
  poisk_st(m,n,a,max,imax);

//вывод матрицы
 okno(10,10,65,25,RED,WHITE,"Результат");
 cprintf("\n\r Максимальная сумма элементов строки (%i) содержится",max);
 cprintf("\n\r в %i-ой строке исходного массива\n\r", imax + 1);
 outmatr(m,n,a,imax);
 cprintf("\n\r Для завершения нажмите <Enter>");

 getchar();
 getchar();
}
```

```
//ввод исходных данных
void inputmatr(int matr[][RAZ], int &str, int &sto)
{int i, j;
 cprintf("\n\r Введите число строк    в массиве <%i: ",RAZ);
 scanf("%i", &str);
 cprintf("\r Введите число столбцов в массиве <%i: ",RAZ);
 scanf("%i", &sto);
 randomize();
 for(i = 0; i < str; i++)        //перебор строк
    for(j = 0; j < sto; j++)      //перебор столбцов
       do{matr[i][j] = random(100);
          }
       while (exist(matr,sto,matr[i][j],i,j));
}

void okno(int x1,int y1,int x2,int y2,int bkcol,int colb,char zag[15])
{window(x1,y1,x2,y2);
 textbackground(bkcol);
 textcolor(colb);
 clrscr();
 gotoxy((x2 - x1 - strlen(zag)) / 2,1);
 cprintf("%s",zag);
}

int exist(int matr[][RAZ],int n,int x,int p,int k)
{int i,j;
 for(i = 0 ; i <= p ; i++)
    for(j = 0 ; j < n ; j++)
       {if((i == p) && (j == k))
         return 0;
     if(matr[i][j] == x)
       return 1;
       }
 return 0;
}

void poisk_st(int str,int sto,int matr[][RAZ],int &max,int &imax)
{int i, j, pr;
 imax = 0;     //за максимум принимаем сумму элементов первой строки
 max = 0;
 for(j = 0; j < sto; j++)
    max += matr[0][j];
 for(i = 1; i < str; i++)
    {pr = 0;
     for(j = 0; j < sto; j++)
     pr += matr[i][j];
     if(max < pr)
       {imax = i;
     max = pr;
       }
    }
}

void outmatr(int m,int n,int matr[][RAZ],int imax)
{int i, j;

 for(i = 0; i < m; i++)
    {for(j = 0; j < n; j++)
```

```
      if(i == imax)
        {textbackground(WHITE);
         textcolor(BLUE);
         cprintf("%4i",matr [i][j]);
         textcolor(WHITE);
         textbackground(RED);
        }
      else cprintf("%4i",matr[i][j]);
     cprintf("\n\r");
    }
}
```

## Лабораторная работа №7
## Методы сортировки

***Цель лабораторной работы:*** *изучение методов сортировки статических структур данных; совершенствование навыков структурного программирования на языке С/С++ при решении задач сортировки матриц.*

***Задание на программирование:*** *используя технологию структурного программирования, реализовать заданный метод сортировки и применить его для указанных фрагментов числовой матрицы в соответствии с индивидуальным заданием.*

***Порядок выполнения работы:***

1) Получить у преподавателя индивидуальное задание: *метод сортировки* и *вид сортируемых фрагментов матрицы*. Исходная матрица не должна содержать одинаковых и нулевых элементов. Значения элементов матрицы необходимо формировать программно (*с клавиатуры не вводить*). Использовать *оконный интерфейс* предыдущей лабораторной работы.

2) Разработать *математическую модель*: описать с помощью формул и рисунков структуру матрицы и процесс её преобразования. У результирующей матрицы должны быть отсортированы заданные фрагменты, а значения элементов не сортируемых фрагментов должны быть обнулены.

3) Построить *схему алгоритма* решения задачи.

4) Составить *спецификации функций*: создания матрицы, вывода матрицы, сортировки заданных фрагментов матрицы, обнуления значений элементов не сортируемых фрагментов матрицы и др.

5) Составить программу на языке С/С++.

6) Проверить и продемонстрировать преподавателю работу программы на *полном* наборе тестов, в том числе с ошибочными входными данными. Обеспечить *одновременный показ* в окнах на экране входной и выходной матриц в одном и том же формате.

7) Оформить *отчет о лабораторной работе* в составе: *постановка задачи, математическая модель, схема алгоритма решения, спецификация функций, текст программы, контрольные примеры*.

***Варианты индивидуальных заданий***

## Методы сортировки

**1**
Сортировка по возрастанию методом выбора минимума.

**2**
Сортировка по возрастанию методом выбора максимума.

**3**
Сортировка по убыванию методом выбора минимума.
**4**
Сортировка по убыванию методом выбора максимума.

**5**
Сортировка по возрастанию методом обмена без флага перестановки.

**6**
Сортировка по убыванию методом обмена без флага перестановки.

**7**
Сортировка по возрастанию методом обмена с флагом перестановки.

**8**
Сортировка по убыванию методом обмена с флагом перестановки.

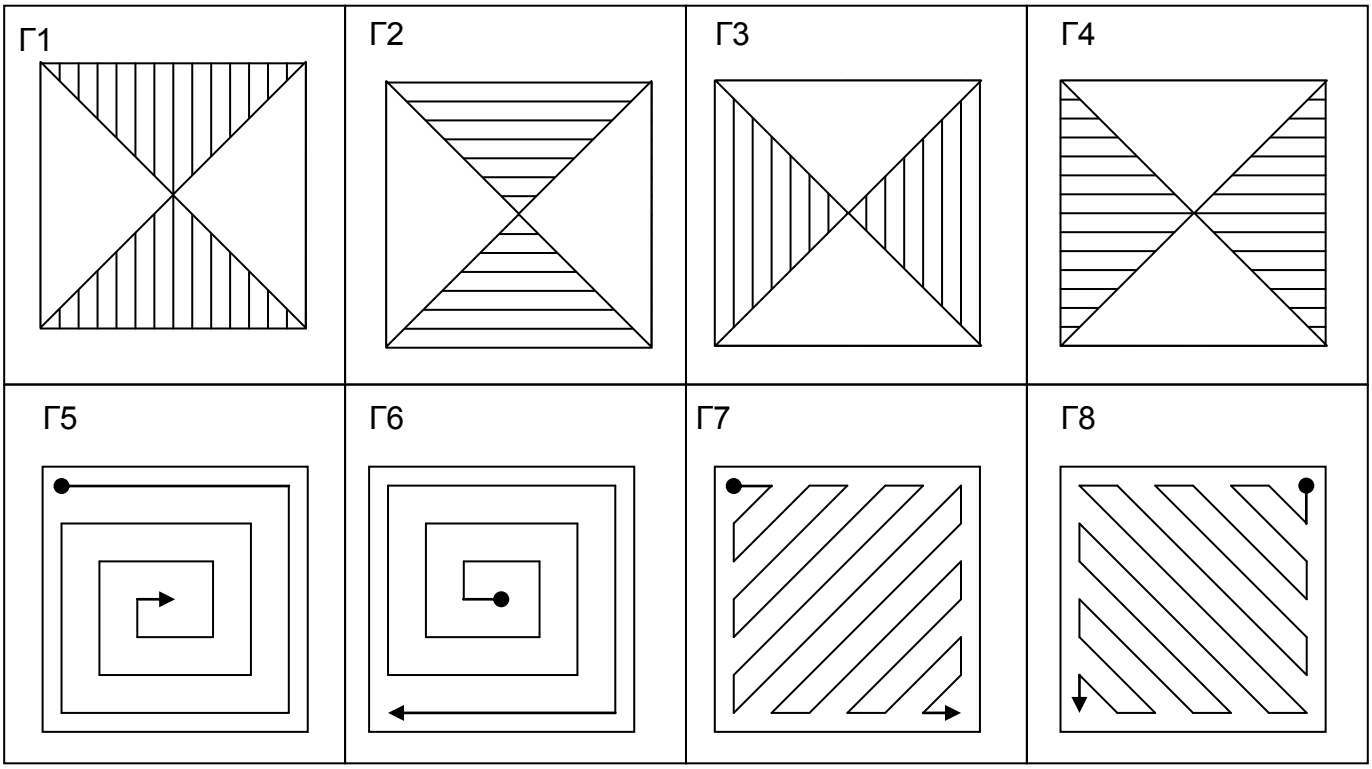
**9**
Сортировка по возрастанию методом вставки.

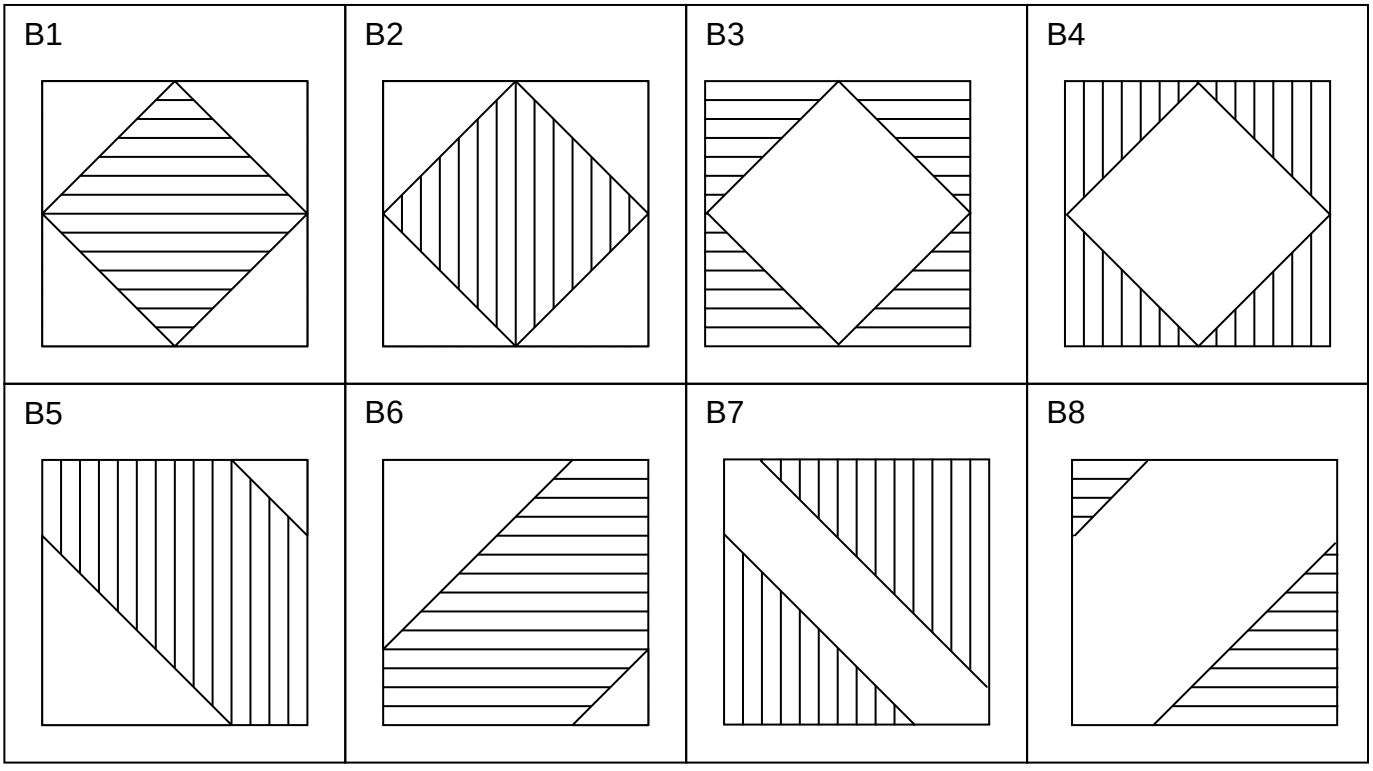
**10**
Сортировка по убыванию методом вставки.

**11**
Быстрая сортировка по возрастанию.

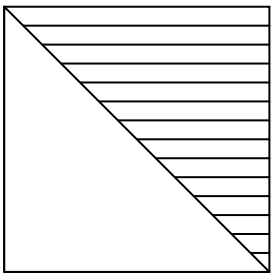
**12**
Быстрая сортировка по убыванию.

## *Сортируемые фрагменты матриц*

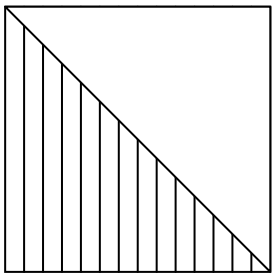


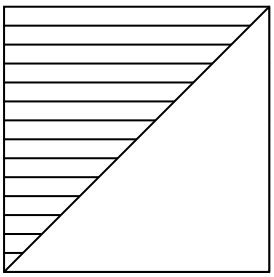
Б1



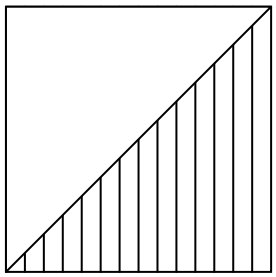
Б2



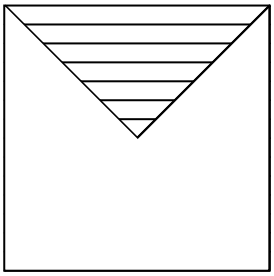
Б3



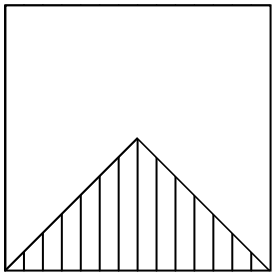
Б4



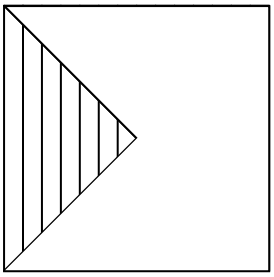
Б5



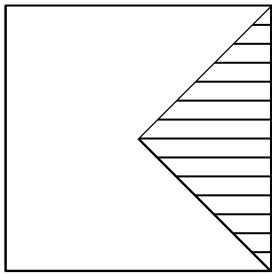
Б6



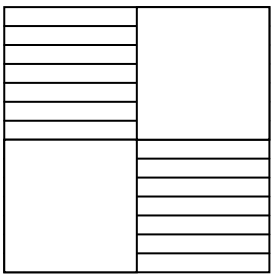
Б7



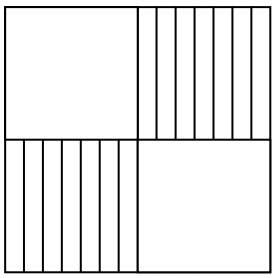
Б8



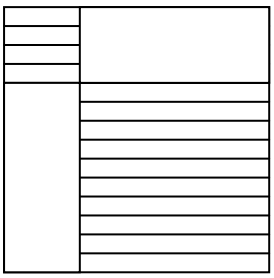
А1



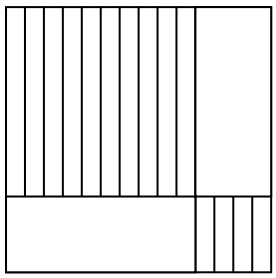
А2



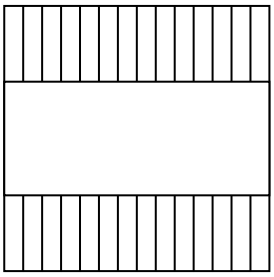
А3



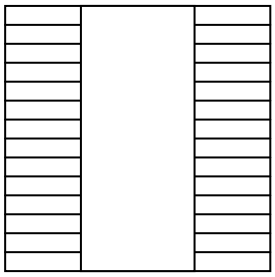
А4



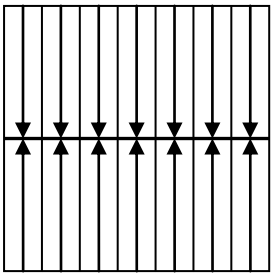
А5



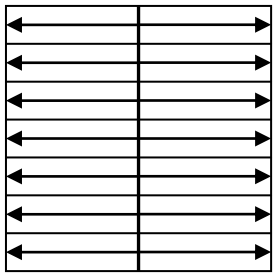
А6



А7



А8



### *Пример программы*

```
//Область 5 (правая половина матрицы ниже главной диагонали).
//Метод вставки. Строки по возрастанию.
#include<stdio.h>
#include<iostream.h>
#include<stdlib.h>
#include<conio.h>
#include<string.h>

const RAZ = 8;              //максимально возможный размер матрицы

void okno(int x1,int y1,int x2,int y2,int bkcol,int colb,char zag[15]);
void obnul(const int n,int**);
void sortir(const int n,int**);
void imputmas(const int n,int a,int b,int**);
void outputmas(const int n,int**);

void main()
{int i,a,b,n;
 int matr[RAZ][RAZ];
 okno(1,1,80,25,BLACK,WHITE,"");
 okno(1,1,30,12,WHITE,BLACK,"Описание");
 cprintf("\r\nВ двумерном массие размером\r\n");
 cprintf("2n x 2n отсортировать строки\r\n");
 cprintf("области № 5 по возрастанию\r\n");
 cprintf("методом вставки.\r\n");
 okno(32,1,79,12,BLUE,WHITE,"Исходная матрица");
 okno(32,14,79,24,BLUE,WHITE,"Результирующая матрица");
 okno(1,14,30,24,WHITE,BLACK,"Окно ввода");
 cout << "\nВведите границы диапазона\n";
 cout << "изменения случайных чисел\n";
 cin >> a >> b;
 cout << "Введите размер матрицы <=4:\n ";
 cin >> n;
 int *dinamo[RAZ];
 for(i = 0; i < RAZ; i++)
    dinamo[i] = matr [i];
 imputmas(n,a,b,dinamo);
 okno(32,1,79,12,BLUE,WHITE,"Исходная матрица");
 outputmas(n,dinamo);
 sortir(n,dinamo);
 obnul(n,dinamo);
 okno(32,14,79,24,BLUE,WHITE,"Результирующая матрица");
 outputmas(n,dinamo);
 getchar();
}

void okno(int x1,int y1,int x2,int y2,int bkcol,int colb,char zag[15])
{window(x1,y1,x2,y2);
 textbackground(bkcol);
 textcolor(colb);
 clrscr();
 gotoxy((x2 - x1 - strlen(zag)) / 2,1);
 cprintf("%s\n\r",zag);
}
```

```
void imputmas(const int n,int a,int b,int** dinamit)
{int i,j;
 randomize();
 for(i = 0; i < 2 * n; i++)
    for(j = 0; j < 2 * n; j++)
       dinamit[i][j] = random(1+b-a) + a;
}

void outputmas(const int n,int** dinamit)
{int i,j;
 for(i = 0; i < 2 * n; i++)
    {for(j = 0; j < 2 * n; j++)
     if((j <= i)&&(j > n - 1))
       {textbackground(WHITE);
        textcolor(RED);
        cprintf("%4i",dinamit[i][j]);
        textcolor(WHITE);
        textbackground(BLUE);
       }
     else cprintf("%4i",dinamit[i][j]);
     gotoxy(1,i+3);
    }
}

void sortir(const int n,int** dinamit)
{int i,j,k,t,b;
 for(k = n + 1; k < 2 * n; k++)            //номер строки
    for(i = n + 1; i <= k; i++)            //номер прохода
       {t = dinamit[k][i];
     b = n;
     while(b < i && dinamit[k][b] <= dinamit[k][i])
          b++;
     for(j = i - 1; j >= b; j--)           //номер столбца
        dinamit[k][j + 1] = dinamit[k][j];
     dinamit[k][b] = t;
       }
}

void obnul(const int n,int** dinamit)
{int i,j;
 for(i = 0; i < 2 * n; i++)
    for(j = 0; j < 2 * n; j++)
       if(!(j <= i && j > n - 1))
      dinamit[i][j] = 0;
}
```

## Лабораторная работа №8
## Обработка строк

***Цель лабораторной работы:*** *изучение стандартных средств языка С/С++ для работы со строками; совершенствование навыков структурного программирования на языке С/С++ при решении задач обработки строк.*

***Задание на программирование:*** *используя технологию структурного программирования разработать программу обработки строки, содержащей не более 80 символов, в соответствии с индивидуальным заданием.*

***Порядок выполнения работы:***

1) Получить у преподавателя индивидуальное задание на обработку строки.
2) Построить *схему алгоритма* решения задачи обработки строки.
3) Составить *спецификации функций*.
4) Составить программу на языке С/С++.
5) Проверить и продемонстрировать преподавателю работу программы на *полном* наборе тестов. Обеспечить *одновременный показ* на экране исходной и отредактированной строк.
6) Оформить *отчет о лабораторной работе* в составе: *постановка задачи, математическая модель, схема алгоритма решения, спецификация функций, текст программы, контрольные примеры.*

### *Варианты индивидуальных заданий*

**1**

Дана строка. Словом текста является последовательность букв русского алфавита; между соседними словами - не менее одного пробела, за последним словом – точка. Найти и сохранить в строке те слова, в которых гласные буквы алфавита образуют симметричную последовательность букв (палиндром). Все остальные слова удалить. Малые и большие буквы алфавита считать эквивалентными.

**2**

Дана строка. Словом текста является последовательность цифр; между соседними словами - не менее одного пробела, за последним словом – точка. Найти и сохранить в строке те слова, в которых все четные цифры образуют неубывающую последовательность чисел. Все остальные слова удалить. Одну цифру не считать неубывающей последовательностью.

**3**

Дана строка. Словом текста является последовательность цифр и букв латинского алфавита; между соседними словами - не менее одного пробела, за последним словом – точка. Найти и сохранить в строке те слова, в которых цифры и буквы латинского алфавита чередуются. Все остальные слова удалить.

**4**

Дана строка. Словом текста считается любая последовательность цифр и букв латинского алфавита; между соседними словами - не менее одного пробела, за последним словом – точка. Найти и сохранить в строке те слова текста, в которых есть хотя бы одна цифра. Все остальные слова удалить.

**5**

Дана строка. Словом текста считается любая последовательность букв русского алфавита; между соседними словами - не менее одного пробела, за последним словом – точка. Найти и сохранить в строке те слова текста, которые содержат только большие буквы алфавита. Все остальные слова удалить.

**6**

Дана строка. Словом текста считается любая последовательность цифр; между соседними словами - не менее одного пробела, за последним словом – точка. Найти и сохранить в строке те слова текста, которые образованы неубывающей последовательностью символов. Все остальные слова удалить.

**7**
Дана строка. Словом текста считается любая последовательность букв русского алфавита; между соседними словами - не менее одного пробела, за последним словом – точка. Найти и сохранить в строке те слова, символы которых образуют симметричную последовательность букв (палиндром). Все остальные слова удалить. Большие и малые буквы алфавита считать эквивалентными.

**8**
Дана строка. Словом текста считается любая последовательность букв русского алфавита; между соседними словами - не менее одного пробела, за последним словом – точка. Удалить из строки те слова, которые содержат двойные согласные буквы.

**9**
Дана строка. Словом текста считается любая последовательность цифр; между соседними словами - не менее одного пробела, за последним словом – точка. Поменять местами в строке первое и последнее слово.

**10**
Дана строка. Словом текста считается любая последовательность букв русского алфавита; между соседними словами - не менее одного пробела, за последним словом – точка. Найти и сохранить в строке те слова текста, которые содержат одинаковое количество гласных и согласных букв алфавита. Все остальные слова удалить.

**11**
Дана строка. Словом текста считается любая последовательность букв русского алфавита; между соседними словами - не менее одного пробела, за последним словом – точка. Найти и сохранить в строке те слова текста, количество гласных букв в которых превышает количество согласных. Все остальные слова удалить.

**12**
Дана строка. Словом текста считается любая последовательность букв латинского алфавита; между соседними словами - не менее одного пробела, за последним словом – точка. Найти и сохранить в строке те слова, которые начинаются с прописной буквы. Все остальные слова удалить.

**13**
Дана строка. Словом текста считается любая последовательность букв русского алфавита; между соседними словами - не менее одного пробела, за последним словом – точка. Найти и сохранить в строке те слова последовательности, которые отличны от последнего слова и являются симметричными. Все остальные слова удалить.

**14**
Дана строка. Словом текста считается любая последовательность букв латинского алфавита; между соседними словами - не менее одного пробела, за последним словом – точка. Найти и сохранить в строке те слова последовательности, которые отличны от первого слова и удовлетворяют следующему свойству: первая буква слова входит в него еще один раз. Все остальные слова удалить.

**15**
Дана строка. Словом текста считается любая последовательность букв латинского алфавита; между соседними словами - не менее одного пробела, за последним словом – точка. Найти и сохранить в строке те слова последовательности, которые отличны от последнего слова и удовлетворяют следующему свойству: слово совпадает с начальным отрезком латинского алфавита (a, ab, abc, abcd,…). Все остальные слова удалить.

**16**
Дана строка. Словом текста считается любая последовательность букв латинского алфавита; между соседними словами - не менее одного пробела, за последним словом – точка. Найти и сохранить в строке те слова последовательности, которые отличны от первого слова и удовлетворяют следующему свойству: слово совпадает с конечным отрезком латинского алфавита (z, yz, xyz,…). Все остальные слова удалить.

**17**
Дана строка. Словом текста считается любая последовательность букв русского алфавита; между соседними словами - не менее одного пробела, за последним словом – точка. Найти и сохранить в строке те слова последовательности, которые отличны от последнего слова и удовлетворяют следующему свойству: в слове нет повторяющихся букв. Все остальные слова удалить.

**18**
Дана строка. Словом текста считается любая последовательность букв латинского алфавита; между соседними словами - не менее одного пробела, за последним словом – точка. Найти и сохранить в строке те слова последовательности, которые отличны от первого слова и удовлетворяют следующему свойству: каждая буква входит в слово не менее двух раз. Все остальные слова удалить.

**19**
Дана строка. Словом текста считается любая последовательность букв русского алфавита; между соседними словами - не менее одного пробела, за последним словом – точка. Найти и сохранить в строке те слова последовательности, которые отличны от последнего слова и удовлетворяют следующему свойству: в слове гласные буквы чередуются с согласными. Все остальные слова удалить.

**20**
Дана строка. Словом текста считается любая последовательность букв латинского алфавита; между соседними словами - не менее одного пробела, за последним словом – точка. Найти и сохранить в строке те слова последовательности, которые отличны от первого слова, предварительно преобразовав каждое из них по следующему правилу: перенести первую букву в конец слова. Все остальные слова удалить.

**21**
Дана строка. Словом текста считается любая последовательность букв русского алфавита; между соседними словами - не менее одного пробела, за последним словом – точка. Найти и сохранить в строке те слова последовательности, которые отличны от последнего слова, предварительно преобразовав каждое из них по следующему правилу: перенести последнюю букву в начало слова. Все остальные слова удалить.

**22**
Дана строка. Словом текста считается любая последовательность букв латинского алфавита; между соседними словами - не менее одного пробела, за последним словом – точка. Найти и сохранить в строке те слова последовательности, которые отличны от первого слова, предварительно преобразовав каждое из них по следующему правилу: удалить из слова первую букву. Все остальные слова удалить.

**23**
Дана строка. Словом текста считается любая последовательность букв русского алфавита; между соседними словами - не менее одного пробела, за последним словом – точка. Найти и сохранить в строке те слова последовательности, которые отличны от последнего слова, предварительно преобразовав каждое из них по следующему правилу: удалить из слова последнюю букву. Все остальные слова удалить.

**24**
Дана строка. Словом текста считается любая последовательность букв латинского алфавита; между соседними словами - не менее одного пробела, за последним словом – точка. Найти и сохранить в строке те слова последовательности, которые отличны от первого слова, предварительно преобразовав каждое из них по следующему правилу: удалить из слова все последующие вхождения первой буквы. Все остальные слова удалить.

**25**
Дана строка. Словом текста считается любая последовательность букв русского алфавита; между соседними словами - не менее одного пробела, за последним словом – точка. Найти и сохранить в строке те слова последовательности, которые отличны от последнего слова, предварительно преобразовав каждое из них по следующему правилу: удалить из слова все предыдущие вхождения последней буквы. Все остальные слова удалить.

**26**
Дана строка. Словом текста считается любая последовательность букв латинского алфавита; между соседними словами - не менее одного пробела, за последним словом – точка. Найти и сохранить в строке те слова последовательности, которые отличны от первого слова, предварительно преобразовав каждое из них по следующему правилу: оставить в слове только первые вхождения каждой буквы. Все остальные слова удалить.

**27**
Дана строка. Словом текста считается любая последовательность букв русского алфавита; между соседними словами - не менее одного пробела, за последним словом – точка. Найти и сохранить в строке те слова последовательности, которые отличны от последнего слова, предварительно преобразовав каждое из них по следующему правилу: если слово нечетной длины, то удалить его среднюю букву. Все остальные слова удалить.

**28**
Дана строка. Словом текста считается любая последовательность букв латинского алфавита; между соседними словами - не менее одного пробела, за последним словом – точка. Разместить в строке последовательность ее слов в обратном порядке.

**29**
Дана строка. Словом текста считается любая последовательность букв русского алфавита; между соседними словами - не менее одного пробела, за последним словом – точка. Найти и сохранить в строке те слова, перед которыми в последовательности находятся только меньшие (по алфавиту) слова, а за ними только большие. Все остальные слова удалить.

**30**
Дана строка. Словом текста считается любая последовательность букв латинского алфавита; между соседними словами - не менее одного пробела, за последним словом – точка. Сохранить в строке последовательность слов, удалив из нее повторные вхождения слов.

**31**

Дана строка. Словом текста считается любая последовательность букв русского алфавита; между соседними словами - не менее одного пробела, за последним словом – точка. Найти и сохранить в строке те слова, которые встречаются в последовательности по одному разу. Все остальные слова удалить.

**32**

Дана строка. Словом текста считается любая последовательность букв латинского алфавита; между соседними словами - не менее одного пробела, за последним словом – точка. Расставить слова строки в алфавитном порядке.

### *Пример программы*

```
//Ввести строку. Вывести слова в алфавитном порядке.
//Функция сравнения строк - стандартная.
#include <stdio.h>
#include <conio.h>
#include <string.h>
#include <stdlib.h>
#include <ctype.h>
const int RAZ = 80;                    //максимальная длина строки
void sort(int n, unsigned char mas[RAZ/2][RAZ]);
int vid_slov(unsigned char isx[RAZ], unsigned char slova[RAZ/2][RAZ]);
void output(int n, unsigned char slova[RAZ/2][RAZ]);
void propis(unsigned char isx[RAZ]);

void main()
{unsigned char st[RAZ];                //исходная строка, м.б. с кириллицей
 unsigned char slova[RAZ / 2][RAZ];    //массив выделенных слов
 int n;                                //число найденных слов

 clrscr();
 printf("Введите строку\n");
 gets(st);                             //функция вводит всю строку, включая
                                       //пробелы и символ /n


 propis(st);                           //преобразуем все буквы в прописные

n = vid_slov(st, slova);               //выделяем в строке отдельные слова

sort(n, slova);                        //сортируем слова по алфавиту

 output(n, slova);                     //выводим слова по алфавиту

 printf("\nДля окончания работы нажмите Enter->");
 getchar();
}

//сортировка слов по алфавиту
void sort(int n, unsigned char slovo[RAZ/2][RAZ])
{int i = 0, fl = 1;
 unsigned char pr[RAZ];

 while(fl)
 {fl = 0;
  i = 0;
  while(i < n - 1)
    {if(strcmp(slovo[i], slovo[i + 1]) > 0)
       {strcpy(pr, slovo[i]);
       strcpy(slovo[i], slovo[i + 1]);
       strcpy(slovo[i + 1], pr);
       fl = 1;
       }
     i++;
    }
 }
}
```

```
//выделяем из исходной строки слова и формируем из них массив
int vid_slov(unsigned char st[RAZ], unsigned char slova[RAZ / 2][RAZ])
{int i = 0, j = 0;

 while(st[i])
   {int k = 0;
    while(st[i] == ' ')
      i++;
    while(st[i] != ' ' && st[i])
      {slova[j][k] = st[i];
       k++;
       i++;
      }
    slova[j][k] = '\0';
    j++;
  }
 return j;
}

//вывод слов на экран
void output(int n, unsigned char slova[RAZ/2][RAZ])
{int i = 0;

 printf("\n");
 while(i < n)
   {puts(slova[i]);
    i++;
   }
}

//перевод строчных букв в прописные
void propis(unsigned char st[RAZ])
{int i=0;

 while(st[i])
   {if(st[i] >= 'a' && st[i] <= 'z' || st[i] >= 'а' && st[i] <= 'п')
      st[i] -= 32;
    else if(st[i] >= 'р' && st[i] <= 'я')
         st[i] -= 80;
    i++;
   }
}
```

## Лабораторная работа №9
## Текстовые файлы

***Цель лабораторной работы:*** *изучение структурной организации, способов доступа к элементам и других особенностей текстовых файлов; изучение стандартных средств языка С/С++ для работы со строками и текстовыми файлами; совершенствование навыков структурного программирования на языке С/С++ при решении задач редактирования текстовых файлов.*

***Задание на программирование:*** *используя технологию структурного программирования разработать программу обработки текстовых файлов с числом строк не менее пяти, каждая из которых содержит не более 80 символов, в соответствии с индивидуальным заданием.*

***Порядок выполнения работы:***

1) Получить у преподавателя индивидуальное задание на обработку строк текстового файла.

2) Построить *схему алгоритма* решения задачи обработки строки.

3) Использовать функции обработки строки, функции создания, просмотра и редактирования текстового файла.

4) Составить *спецификации функций*.

5) Составить программу на языке С/С++.

6) Проверить и продемонстрировать преподавателю работу программы на *полном* наборе тестов. Обеспечить *одновременный показ* на экране исходной и отредактированной строк.

7) Оформить *отчет о лабораторной работе* в составе: *постановка задачи, математическая модель, схема алгоритма решения, спецификация функций, текст программы, контрольные примеры.*

### *Пример программы*

```
//Программа создает файл строк. Признак окончания ввода - ввод пустой строки.
//Слово - это последовательность русских букв. Между словами не менее одного
//пробела. Затем строки считываются и программа печатает те слова из каждой
//строки, которые содержат равное количество гласных и согласных букв.

#include <stdio.h>
#include <conio.h>
#include <string.h>
#include <stdlib.h>

const FNAME = "C:\\stroka.txt\0";     //имя файла объявлено константой
void make_file(char *fname);          //создание файла
void pro_verka(char *fname);          //чтение и обработка строк файла
int glasn(char ch);                   //проверка на гласность
int so_glasn(char ch);                //проверка на согласность

void main()
{clrscr();
 char fname[20] = FNAME;
 make_file(fname);
 pro_verka(fname);
}

//Функция проверяет, является ли символ гласной русской буквой
int glasn(char ch)
{static char gl[] = "АаЕеИиОоУуЫыЭэЮюЯя\0";
 int i = 0;

 while(gl[i] && gl[i] != ch)
      i++;
 if(gl[i])
   return(1);                         //значит буква - гласная
 else return(0);                      //значит буква - не гласная
}

//Функция проверяет, является ли символ согласной русской буквой
int so_glasn(char ch)
{static char so_gl[] = "БбВвГгДдЖжЗзКкЛлМмНнПпРрСсТтФфХхЦцЧчШшЩщ\0";
 int i = 0;

 while(so_gl[i] && so_gl[i] != ch)
      i++;
 if(so_gl[i])
   return(1);                         //значит буква - согласная
 else return(0);                      //значит буква - не согласная
}

//Функция создания текстового файла
void make_file(char *fname)
{unsigned char st[80];                //исходная строка
 FILE *in;                            //текстовый файл
 puts("\nСоздание файла");
 puts("После ввода каждой строки нажмите <Enter>.");
 puts("Признак окончания ввода - ввод пустой строки\n");
```

```
//Открываем файл в режиме записи (w) текста (t)
//Если файл с таким именем уже есть, то новые данные
//будут дописаны поверх старых
 if((in = fopen(fname, "wt")) == NULL)
   {printf("Ошибка открытия файла для записи. Нажмите <Enter>");
    getchar();
    exit(0);
   }

 printf("Введите строку и нажмите <Enter>\n");
 printf("->");
 gets(st);                  //функция вводит всю строку, включая
                            //пробелы и символ \n
 while(strlen(st) != 0)
   {fprintf(in, "%s\n", st);
    printf("\nВведите строку и нажмите <Enter>\n");
    printf("->");
    gets(st);
   }
 fclose(in);                //закрываем файл
}

//читаем и обрабатываем строки файла
void pro_verka(char *fname)
{FILE *in;                              //текстовый файл
 unsigned char st[80];                  //исходная строка
 unsigned char sr[80];                  //результирующая строка
 unsigned char pr[80];                  //обрабатываемое слово
 int i,j,k;                             //номер обрабатываемого символа
 int gl = 0;                            //число гласных букв в слове
 int sogl = 0;                          //число согласных букв в слове
 int ok;                                //признак гласной (согласной) буквы
 int n;                                 //длина результирующей строки
 int m;                                 //максимально возможная длина строки
//Открываем файл в режиме чтения (r) текста (t)
 if((in = fopen(fname, "rt")) == NULL)
   {printf("Ошибка открытия файла для чтения");
    getchar();
    exit(0);
   }

 printf("\nРезультат:\n");
 m = 80;
 fgets(st,m,in);                        //читаем строку файла
 while(!feof(in))
   {i = 0;
    n = 0;
    sr[i] = '\0';

    while(st[i])                        //обрабатываем строку
      {k = 0;
       while(st[i] != ' ' && st[i+1]) //копируем слово
       {pr[k] = st[i];
        k++;
        i++;
       }
       pr[k] = '\0';
```

```
       j = 0;                          //обрабатываем слово
       gl = 0;
       sogl = 0;
       while(pr[j])
       {ok = glasn(pr[j]);             //считаем гласные
        if(ok)
          gl++;

        ok = so_glasn(pr[j]);          //считаем согласные
        if(ok)
          sogl++;
        j++;
       }
        if(gl == sogl)
       {for(j = 0; j < k; j++,n++)
            sr[n] = pr[j];
        if(st[i])
          sr[n] = ' ';
        n++;
       }
        i++;
      }

    sr[n] = '\0';
    printf("\n%s",sr);
    fgets(st,m,in);
   }
 fclose(in);                           //Закрываем файл
 printf("\nДля завершения нажмите <Enter>");
 getch();
}
```

## Лабораторная работа №10
## Базы данных

***Цель лабораторной работы:*** *изучение структурной организации, способов доступа к элементам и других особенностей файлов структур; изучение стандартных средств языка С/С++ для работы с файлами; совершенствование навыков структурного программирования на языке С/С++ при решении задач обработки файлов.*

***Задание на программирование:*** *используя технологию структурного программирования разработать программу обработки файлов структур с числом записей не менее пяти в соответствии с индивидуальным заданием.*

***Порядок выполнения работы:***

1) Получить у преподавателя индивидуальное задание.
2) Построить *схему алгоритма* решения задачи.
3) Сформулировать условие поиска данных в файле и организовать поиск по условию с сохранением найденных записей в новом файле.
4) Использовать функции создания, просмотра, сортировки файла, поиска данных в файле.
5) Составить *спецификации функций*.
6) Составить программу на языке С/С++.
7) Предусмотреть в программе возможность выбора варианта действия с помощью меню в окне диалога с пользователем.
8) Проверить и продемонстрировать преподавателю работу программы на *полном* наборе тестов. Обеспечить *одновременный показ* на экране исходного и результирующего файла.
9) Оформить *отчет о лабораторной работе* в составе: *постановка задачи, математическая модель, схема алгоритма решения, спецификация функций, текст программы, контрольные примеры.*

## *Варианты индивидуальных заданий*

**1** Самолеты

| Наименование типа | Фамилия конструктора | Год выпуска | Количество кресел | Грузоподъемность в Т |
|---|---|---|---|---|

**2** Расчет движения

| Наименование воздушной линии | Тип самолета | Количество рейсов | Налет в тыс.км | Пассажирооборот в чел. км |
|---|---|---|---|---|

**3** Перевозки

| Тип самолета | Номер борта | Количество рейсов | Налет в часах | Налет в тыс.км |
|---|---|---|---|---|

**4** Расписание

| Номер Рейса | Наименование рейса | Тип самолета | Стоимость билета | Протяженность линии |
|---|---|---|---|---|

**5** Сооружения аэропорта

| Наименование | Площадь | Этажность | Год сооружения | Стоимость в млн. руб. |
|---|---|---|---|---|

**6** Ремонт аэродромных сооружений

| Наименование | Шифр | Вид ремонта | Стоимость ремонта | Наименование подрядчика |
|---|---|---|---|---|

**7** Кассы авиабилетов

| Номер кассы | ФИО кассира | Количество проданных билетов | Суммарная выручка | Дата продажи |
|---|---|---|---|---|

**8** Характеристики ПК

| Тип процессора | Тактовая частота | Емкость ОП в Мбайт | Емкость ЖМД в Мбайт | Тип монитора |
|---|---|---|---|---|

**9** Города

| Наименование | Количество жителей | Площадь в кв.км | Год основания | Количество школ |
|---|---|---|---|---|

**10** Мосты

| Наименование | Высота | Ширина | Количество опор | Протяженность |
|---|---|---|---|---|

**11** Программные продукты

| Наименование | Фирма | Стоимость | Объем | Количество |
|---|---|---|---|---|

**12** Музеи

| Наименование | Назначение | Адрес | Время работы | Стоимость билета |
|---|---|---|---|---|

**13** Автоинспекция

| Марка машины | Цвет | Гос. номер | Год выпуска | Владелец |
|---|---|---|---|---|

**14** Квартиры

| Адрес | Площадь в кв.м | Сторона света | Стоимость 1 кв.м | Этаж | Колич. комнат |
|---|---|---|---|---|---|

**15** Кинотеатры

| Наименование | Стоимость билета | Время сеансов | Адрес мест | Количество мест |
|---|---|---|---|---|

**16** Магазин

| Наименование товара | Фирма изготовитель | Сорт | Цена | Размер партии |
|---|---|---|---|---|

**17** Театр

| Наименование спектакля | Дата | Время | Место | Цена билета |
|---|---|---|---|---|

**18** Железная дорога

| Пункт назначения | Поезд | Вагон | Место | Стоимость проезда |
|---|---|---|---|---|

**19** Библиотека

| Название книги | Автор | Издание | Год издания | Количество экземпляров |
|---|---|---|---|---|

**20** Экскурсии

| Наименование | Страна | Стоимость | Продолжительность | Транспорт |
|---|---|---|---|---|

**21** Метрополитен

| Номер линии | Название линии | Число станций | Время стоянки | Время разворота |
|---|---|---|---|---|

### *Пример программы*

```
//Программа работы с базой данных "Экскурсии"
//Создание базы
//Просмотр базы
//Поиск по названию страны с созданием файла выборки
//Сортировка по наименованию экскурсии в алфавитном порядке
//Сортировка в порядке возрастания стоимости путевки
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>
#include <string.h>
#include <iostream.h>;
#define FNAME1 "A:\\bd1.dat"              //имя файла с исходной базой
#define FNAME2 "A:\\bd2.dat"              //имя файла с результатами поиска
void wind(int x1,int y1,int x2,int y2,int colf,int colb);
void dobavka();
void zag1();
void zag2();
void prosmotrbd1(char *fname);
void prosmotrbd2(char *fname);
void poiskcountry(char *fname1, char *fname2);
void sort_name(char *fname);
void sort_voz_cena(char *fname);
struct trip{unsigned char name[15];       //наименование экскурсии
            unsigned char country[15];    //страна
            unsigned int cena;            //стоимость путевки
            unsigned int time;            //продолжительность
            unsigned char trans[10];      //транспорт
           };
FILE *baza1;
trip excur;
char otv;

int main()
{int var;
 textbackground(BLACK);
 clrscr();
 wind(1, 1, 80, 25, 1, 15);
 if((baza1 = fopen(FNAME1, "r+")) != NULL)
   {printf(" База данных экскурсий была создана раньше.\n");
    printf(" Добавлять новые записи в базу экскурсий? [Y/N]");
    while(otv = getchar() == '\n');
    if(otv == 'Y' || otv == 'y' || otv == 'H' || otv == 'н')
      if((baza1 = fopen(FNAME1, "a")) == NULL)
        {printf("\n Ошибка открытия базы данных для добавления\n");
         abort();
        }
      else{printf("\n Добавляем новые записи\n");
           dobavka();
           fclose(baza1);
          }
   }
 else if((baza1 = fopen(FNAME1, "w+")) == NULL)
        {printf("\n Ошибка открытия пустой базы данных для чтения и записи\n");
         abort();
        }
      else{printf(" Создаем новую базу\n");
           dobavka();
           fclose(baza1);
          }
 if((baza1 = fopen(FNAME1, "r+")) == NULL)
```

```
   {printf("\n Ошибка открытия базы данных для чтения и записи\n");
    abort();
   }
 else printf("\n База данных успешно создана\n");
 printf("\n Для продолжения нажмите Enter->");
 getchar();

 for( ; ;)                          //меню программы
  {wind(1, 1, 80, 25, 0, 15);
   wind(20, 1, 60, 9, 1, 15);
//Выбор вида действия
   cprintf( "\n    Вид действия:\n\r");
   cprintf("  1 - сортировка по наименованию\n\r");
   cprintf("  2 - сортировка по цене путевки\n\r");
   cprintf("  3 - поиск по стране\n\r");
   cprintf("  4 - просмотр базы данных\n\r");
   cprintf("  5 - просмотр базы данных поиска\n\r");
   cprintf("  6 - завершение задачи\n\r");
   cprintf("  Введите вид действия ->");
   cin >> var;
   if(var == 6) break;
   switch(var)
      {case 1: wind(1, 10, 80, 15, 4, 15);
             sort_name(FNAME1);
             printf("\n Сортировка закончена.");
             printf("\n Для продолжения нажмите Enter->");
             getchar();
             break;
       case 2: wind(1, 10, 80, 15, 2, 15);
             sort_voz_cena(FNAME1);
             printf("\n Сортировка закончена.");
             printf("\n Для продолжения нажмите Enter->");
             getchar();
             break;
       case 3: wind(1, 10, 80, 25, 2, 15);
             poiskcountry(FNAME1, FNAME2);
             printf("\n Поиск по стране закончен.");
             printf("\n Для продолжения нажмите Enter->");
             getchar();
             break;
       case 4: wind(1, 10, 80, 25, 2, 15);
             prosmotrbd1(FNAME1);
             printf("\n Для продолжения нажмите Enter->");
             getchar();
             break;
       case 5: wind(1, 10, 80, 25, 2, 15);
             prosmotrbd2(FNAME2);
             printf("\n Для продолжения нажмите Enter->");
             getchar();
      }
 }
  return 0;
}

//Вывод окна на экран
void wind(int x1,int y1,int x2,int y2,int colf,int colb)
{window(x1,y1,x2,y2);
 textbackground(colf);
 textcolor(colb);
 clrscr();
}
```

```
//Добавление новых элементов в базу данных
void dobavka()
{do
  {printf("\nНаименование экскурсии? ");
   scanf("%s", &excur.name);

   printf("\nСтрана? ");
   scanf("%s", &excur.country);

   printf("\nСтоимость путевки? ");
   scanf("%u", &excur.cena);

   printf("\nПродолжительность? ");
   scanf("%u", &excur.time);

   printf("\nТранспорт? ");
   scanf("%s", &excur.trans);

   fwrite(&excur, sizeof(excur), 1, baza1);

   printf("\nПродолжать?[Y/N]");
   while((otv = getchar()) == '\n');
  }
 while(otv == 'Y' || otv == 'y' || otv == 'Н' || otv == 'н');
}

//Вывод заголовка при просмотре исходного файла
void zag1()
{int i ;
 printf("\n");
 for(i = 1; i <= 65; i++)
    printf("-");
 printf("\n|%15s|%15s|%10s|%10s|%10s\n",
      "Наименование","Страна","Стоимость","Продолжит.","Транспорт");
 for(i = 1; i <= 65; i++)
    printf("-");
}

//Вывод заголовка при просмотре файла поиска
void zag2()
{int i ;
 printf("\n");
 for(i = 1; i <= 65; i++)
    printf("-");
 printf("\n|%15s|%15s|%10s|%10s|%10s\n",
      "Страна","Наименование","Стоимость","Продолжит.","Транспорт");
 for(i = 1; i <= 65; i++)
    printf("-");
}

//Просмотр базы данных экскурсий
void prosmotrbd1(char *fname)
{int i ;
 FILE *baza1;
 if((baza1 = fopen(fname, "r+")) == NULL)
   {printf("\n Ошибка открытия базы данных\n");
    abort();
   }
 printf("\n     База данных экскурсий");
 zag1();
 rewind(baza1);
 while(fread(&excur, sizeof(excur), 1, baza1) > 0)
 {printf("\n|%15s|%15s|%10u|%10u|%10s",
```

```
  excur.name, excur.country, excur.cena, excur.time, excur.trans);
 }
 printf("\n");
 for(i = 1; i <= 65; i++)
    printf("-");
}

//Просмотр базы данных поиска экскурсий по стране пребывания
void prosmotrbd2(char *fname)
{int i;
 FILE *baza2;
 if((baza2 = fopen(fname, "r+")) == NULL)
   {printf("\n Ошибка открытия базы данных\n");
    abort();
   }
 printf("\n  База данных поиска экскурсий по стране");
 zag2();
 rewind(baza2);
 while(fread(&excur, sizeof(excur), 1, baza2) > 0)
   {printf("\n|%15s|%15s|%10u|%10u|%10s",
    excur.country, excur.name, excur.cena, excur.time, excur.trans);
   }
 printf("\n") ;
 for(i = 1; i <= 65; i++)
    printf("-");
}

//Поиск по стране пребывания
void poiskcountry(char *fname1, char *fname2)
{unsigned char country[15];
 FILE *baza1,
      *baza2;
 if((baza2=fopen(fname2, "w+")) == NULL)
   {printf("\n Ошибка открытия пустой базы данных для записи\n");
    abort();
   }
 if((baza1 = fopen(fname1, "r+")) == NULL)
   {printf("\n Ошибка открытия базы данных для чтения и записи\n");
    abort();
   }

 printf("\n Названиe страны для поиска? ");
 scanf("%s", &country);
 rewind(baza1);
 while(fread(&excur, sizeof(excur), 1, baza1) > 0)
 if(strncmp(excur.country, country, 15) == 0)
   {fwrite(&excur, sizeof(excur), 1, baza2);
   }
 fclose(baza2);
 fclose(baza1);
 getchar();
}

//Сортировка по наименованию экскурсии по алфавиту
void sort_name(char *fname)
{int i;
 int fl;
 trip ppp;
 FILE *baza1;

 if((baza1 = fopen(fname, "r+")) == NULL)
   {printf("\n Ошибка открытия базы данных для чтения и записи\n");
    abort();
```

```
   }
 fl=0;
 do{rewind(baza1);
    fl=0;
    for(i=0; fread(&excur, sizeof(excur), 1, baza1) > 0; i += sizeof(excur),
      fseek(baza1, i, SEEK_SET))    //позиция i от НАЧАЛА файла
    {if(fread(&ppp, sizeof(excur), 1, baza1) > 0)
       {if(strncmp(excur.name, ppp.name, 15) > 0)
        {fseek(baza1, i, SEEK_SET); //позиция i от НАЧАЛА файла
         fwrite(&ppp, sizeof(excur), 1, baza1);
         fwrite(&excur, sizeof(excur), 1, baza1);

         fl = 1;
        }
       }
    }
   }
 while(fl);
 fclose(baza1);
}

//Сортировка по убыванию стоимости путевки
void sort_voz_cena(char *fname)
{int i;
 int fl;
 trip ppp;
 FILE *baza1;

 if((baza1 = fopen(fname, "r+")) == NULL)
   {printf("\n Ошибка открытия базы данных для чтения и записи\n");
    abort();
   }
 fl = 0;
 do{rewind(baza1);
    fl = 0;
    for(i=0; fread(&excur, sizeof(excur), 1, baza1) > 0; i += sizeof(excur),
      fseek(baza1, i, SEEK_SET))
      {if(fread(&ppp, sizeof(excur), 1, baza1) > 0)
         {if(excur.cena > ppp.cena)
            {fseek(baza1, i, SEEK_SET);  //позиция i от НАЧАЛА файла
             fwrite(&ppp, sizeof(excur), 1, baza1);
             fwrite(&excur, sizeof(excur), 1, baza1);
             fl = 1;
            }
         }
      }
   }
 while(fl);
 fclose(baza1);
}
```

## Лабораторная работа №11
## Линейные списки

***Цель лабораторной работы:*** *изучение способов создания и принципов использования односвязных линейных списков; изучение стандартных средств языка С/С++ для работы с динамической памятью; совершенствование навыков структурного программирования на языке С/С++ при решении задач обработки линейных списков.*

***Задание на программирование:*** *используя технологию структурного программирования разработать программу обработки односвязных линейных списков с числом элементов в списке не менее пяти в соответствии с индивидуальным заданием.*

***Порядок выполнения работы:***

1) Получить у преподавателя индивидуальное задание.
2) Построить *схему алгоритма* решения задачи.
3) Использовать функции создания, просмотра, обработки списка, удаления списка из динамической памяти.
4) Составить *спецификации функций*.
5) Составить программу на языке С/С++.
6) Проверить и продемонстрировать преподавателю работу программы на *полном* наборе тестов. Обеспечить *одновременный показ* на экране исходного и результирующего списка.
7) Оформить *отчет о лабораторной работе* в составе: *постановка задачи, математическая модель, схема алгоритма решения, спецификация функций, текст программы, контрольные примеры.*

## Варианты индивидуальных заданий

**1**
По списку L построить два новых списка L1 и L2: первый из положительных элементов, а второй из остальных элементов списка.

**2**
Вставить в список L новый элемент E1 за каждым вхождением заданного элемента E, если E входит в L.

**3**
Вставить в список L новый элемент E1 перед каждым вхождением элемента E, если E входит в L.

**4**
Вставить в непустой список L перед его последним элементом пару новых элементов E1 и E2.

**5**
Вставить в непустой список L, элементы которого упорядочены по не убыванию, новый элемент E так, чтобы сохранить упорядоченность списка.

**6**
*Удвоить каждое вхождение элемента E в списке L.*

**7**
Удалить из списка L все вхождения элемента E.

**8**
Удалить из списка L все отрицательные элементы.

**9**
Удалить из списка L за каждым вхождением элемента E один элемент, если он есть и отличен от E.

**10**
Оставить в списке L только первые вхождения одинаковых элементов.

**11**
В списке L из каждой группы подряд идущих равных элементов оставить только один.

**12**
Перевернуть список L, то есть изменить ссылки в этом списке так, чтобы его элементы оказались расположенными в обратном порядке.

**13**
Определить, входит ли список L1 в список L2.

**14**
Проверить, есть ли в списке L хотя бы два одинаковых элемента.

**15**
Проверить на равенство два списка L1 и L2.

**16**
Построить список L1 – копию списка L.

**17**
Добавить в конец списка L1 все элементы списка L2.

**18**
Вставить в список L за первым вхождением элемента E все элементы списка L1, если E входит в L.

**19**
Сформировать список L, включив в него по одному разу элементы, которые входят хотя бы в один из списков L1 и L2.

**20**
*Сформировать список L, включив в него по одному разу элементы, которые входят одновременно в оба списка L1 и L2.*

**21**
Сформировать список L, включив в него по одному разу элементы, которые входят в список L1, но не входят в список L2.

**22**
Сформировать список L, включив в него по одному разу элементы, которые входят в один из списков L1 и L2, но в то же время не входят в другой из них.

**23**
*Объединить два упорядоченных списка L1 и L2 в один упорядоченный список, построив новый список L.*

**24**
Объединить два упорядоченных списка L1 и L2 в один упорядоченный список L1, меняя соответствующим образом ссылки в L1 и L2.

**25**
Найти среднее арифметическое элементов непустого списка.

**26**
Поменять местами первый и последний элемент списка.

**27**
Проверить, упорядочены ли элементы списка по алфавиту.

**28**
Найти сумму последнего и предпоследнего элементов списка.

**29**
Вставить в начало списка новый элемент.

**30**
Вставить в конец списка новый элемент.

**31**
Вставить новый элемент после первого элемента непустого списка.

**32**
Удалить из непустого списка первый элемент.

**33**
Удалить из списка второй элемент, если такой есть.

**34**
Удалить из непустого списка последний элемент.

**35**
Удалить из списка первый отрицательный элемент, если такой есть.

**36**
Заменить в списке L все вхождения элемента E1 на E2.

**37**
Перенести в конец списка его первый элемент.

**38**
Перенести в начало списка его последний элемент.

**39**
Определить, входит ли элемент E в список L.

**40**
Подсчитать число вхождений элемента E в список L.

**41**
Найти максимальный элемент непустого списка.

**42**
Удалить из списка L первое вхождение элемента E, если такое есть.

**43**
Подсчитать количество слов списка, которые начинаются и оканчиваются одной и той же литерой.

**44**
Подсчитать количество слов списка, которые начинаются с той же литеры, что и следующее слово.

**45**
Подсчитать количество слов списка, которые совпадают с последним словом.

### *Пример программы*

```
#include <iostream.h>
#include <string.h>
#include <stdio.h>
#include <conio.h>
#include <dos.h>
const int n = 80;
struct spis{char ch[n];
         spis* next;
        };
void okno(int x1,int y1,int x2,int y2,int bkcol,int colb,char zag[15]);
spis* sozdspis(char ch[80],spis* head);
int obrabotka(spis* head);
void udalspis(spis* head);
void viewsp(spis* head);

int main()
{char ch[80];
 int kol;
 spis* head = 0;
 okno(1,1,80,25,BLACK,WHITE,"");
 okno(1,1,32,12,WHITE,BLACK,"Описание");
 cout << "\n Подсчитать количество"
         "\n слов списка L, которые"
         "\n начинаются с той же литеры,"
         "\n что и следующее слово";
 okno(34,1,79,12,BLUE,WHITE,"Исходный список");
 okno(34,14,79,24,BLUE,WHITE,"Результат подсчета");
 okno(1,14,32,24,WHITE,BLACK,"Окно ввода");
 cout << "\n Вводите элементы списка L"
         "\n (слова) через пробел;"
         "\n после последнего слова"
         "\n через пробел - точка";
 cout << "\n ";
 do
   {cin >> ch;
    head = sozdspis(ch,head);
   }
 while(ch[0] != '.');
 okno(34,1,79,12,BLUE,WHITE,"Исходный список");
 gotoxy(2,3);
 viewsp(head);
 getchar();
 kol = obrabotka(head);
 okno(34,14,79,24,BLUE,WHITE,"Результат подсчета");
 gotoxy(2,3);
 cprintf("\r\n Найдено %u таких слов",kol);
 getchar();
 return 0;
}

//Формирование окна диалога
void okno(int x1,int y1,int x2,int y2,int bkcol,int colb,char zag[15])
{window(x1, y1, x2, y2);
 textbackground(bkcol);
 textcolor(colb);
```

```
 clrscr();
 gotoxy((x2 - x1 - strlen(zag)) / 2,1);
 cprintf("%s\n\r",zag);
}

//Добовление нового элемента в список
spis* sozdspis(char ch[n], spis* head)
{spis *tec, *nov;
 nov = new(spis);
 strcpy(nov -> ch,ch);
 nov -> next = 0;
 if(head)             //список не пуст
   {tec = head;
    while(tec -> next)
       tec = tec -> next;
    tec -> next = nov;
   }
 else                 //список пуст
    head = nov;
 return head;
}

//Подсчет числа вхождений
int obrabotka(spis* head)
{spis* tec;
 int kol = 0;
 tec = head;
 while(tec -> next -> next != NULL)
      {if(tec -> next -> ch[0] == tec -> ch[0])
       kol++;
       tec = tec -> next;
      }
 return kol;

}

//Просмотр списка
void viewsp(spis* head)
{spis* tec;
 tec = head;
 while(tec -> next != NULL)
      {cprintf("%s ",tec -> ch);
       tec = tec -> next;
      }
}

//Удаление списка
void udalspis(spis* head)
{spis *pred,*tec;
 tec = head;
 while(tec != NULL)
      {pred = tec;
       tec = tec -> next;
       delete(pred);
       pred = NULL;
      }
}
```

## Лабораторная работа №12
## Динамические структуры данных

***Цель лабораторной работы:*** *изучение способов создания и принципов использования динамических структур данных типа стек, дек, очередь; изучение стандартных средств языка C/C++ для работы с динамической памятью; совершенствование навыков структурного программирования на языке C/C++ при решении задач обработки динамических структур данных.*

***Задание на программирование:*** *используя технологию структурного программирования разработать программу обработки данных, содержащихся в заранее подготовленном файле, в соответствии с индивидуальным заданием. Применить динамическую структуру указанного в задании вида: стек, очередь или дек.*

***Порядок выполнения работы:***

1) Получить у преподавателя индивидуальное задание.
2) Построить *схему алгоритма* решения задачи.
3) Использовать функции, реализующие полный набор операций для этой структуры:
   - допустимые операции ***для стека***: инициализация, проверка пустоты, добавление нового элемента в начало, извлечение элемента из начала;
   - допустимые операции ***для очереди***: инициализация, проверка пустоты, добавление нового элемента в конец, извлечение элемента из начала;
   - допустимые операции ***для дека***: инициализация, проверка пустоты, добавление нового элемента в начало, добавление нового элемента в конец, извлечение элемента из начала, извлечение элемента из конца.
4) Составить *спецификации функций*.
5) Составить программу на языке C/C++.
6) Проверить и продемонстрировать преподавателю работу программы на *полном* наборе тестов. Обеспечить *одновременный показ* в окнах на экране входных и выходных данных.
7) Оформить *отчет о лабораторной работе* в составе: *постановка задачи, математическая модель, схема алгоритма решения, спецификация функций, текст программы, контрольные примеры.*

## Варианты индивидуальных заданий

**1**

Отсортировать строки файла, содержащие названий книг, в алфавитном порядке с использованием двух ***деков*.**

**2**

***Дек*** содержит последовательность символов для шифровки сообщений. Дан текстовый файл, содержащий зашифрованное сообщение. Пользуясь ***деком,*** расшифровать текст. Известно, что при шифровке каждый символ сообщения заменялся следующим за ним в ***деке*** по часовой стрелке через один.

**3**

***Дек*** содержит последовательность символов для шифровки сообщений. Дан текстовый файл, содержащий сообщение. Пользуясь ***деком,*** зашифровать текст, заменяя каждый символ сообщения следующим за ним в ***деке*** против часовой стрелки через один.

**4**

Написать программу, моделирующую железнодорожный сортировочный узел. Исходный файл содержит информацию об имеющихся вагонах двух типов, при этом количество вагонов обоих типов одинаково. Последовательность элементов файла неупорядочена, в каждом элементе файла: тип вагона и идентификационный номер вагона. Используя ***стек*** (“тупик”), за один просмотр исходного файла сформировать новый файл (“состав вагонов”), в котором типы вагонов чередуются.

**5**

Даны три стержня и *n* дисков различного размера. Диски можно надевать на стержни, образуя из них башни. Перенести *n* дисков со стержня *A* на стержень *C***,** сохранив их первоначальный порядок. При переносе дисков необходимо соблюдать следующие правила:

на каждом шаге со стержня на стержень переносить только один диск;

диск нельзя помещать на диск меньшего размера;

для промежуточного хранения можно использовать стержень *B***.**

Реализовать алгоритм, используя три ***стека*** вместо стержней *A***,** *B***,** *C***.** Информация о дисках хранится в исходном файле.

**6**

Дан файл из вещественных чисел. Используя ***очередь,*** за один просмотр файла напечатать сначала все числа, меньшие *a***,** затем все числа из интервала *[a,b]***,** и, наконец, все остальные числа, сохраняя исходный порядок в каждой группе.

**7**
Дан текстовый файл с программой на алгоритмическом языке. За один просмотр файла проверить баланс круглых скобок в тексте, используя ***стек.***

**8**
Дан текстовый файл с программой на алгоритмическом языке. За один просмотр файла проверить баланс круглых скобок в тексте, используя ***очередь.***

**9**
Дан текстовый файл. Используя ***очередь,*** переписать содержимое его строк в новый текстовый файл, перенося при этом в конец каждой строки все входящие в нее цифры, сохраняя исходный порядок следования среди цифр и среди остальных символов строки.

**10**
Дан файл из символов. Используя ***очередь,*** за один просмотр файла напечатать сначала все цифры, затем все буквы, и, наконец, все остальные символы, сохраняя исходный порядок в каждой группе символов.

**11**
Дан текстовый файл. Используя ***стек***, сформировать новый текстовый файл, каждая строка которого содержит символы соответствующей строки исходного файла, записанные в обратном порядке.

**12**
Дан файл из целых чисел. Используя ***очередь,*** за один просмотр файла напечатать сначала все отрицательные числа, затем все положительные числа, сохраняя исходный порядок в каждой группе.

**13**
Дан текстовый файл. Используя ***стек,*** сформировать новый текстовый файл, содержащий строки исходного файла, записанные в обратном порядке: первая строка становится последней, вторая – предпоследней и т.д.

**14**
Дан текстовый файл. Используя ***очередь,*** переписать содержимое его строк в новый текстовый файл, перенося при этом в начало каждой строки все входящие в нее буквы, затем все цифры, и, наконец, все остальные символы строки, сохраняя исходный порядок в каждой группе символов.

**15**

Дан текстовый файл. Используя ***стек,*** вычислить значение логического выражения, записанного в текстовом файле в следующей форме:

< ЛВ > ::= **T** | **F** | (**N**<ЛВ>) | (<ЛВ>**A**<ЛВ>) | (<ЛВ>**X**<ЛВ>) | (<ЛВ>**O**<ЛВ>),

где буквами обозначены логические константы и операции:

**T** – True, **F** – False, **N** – Not, **A** – And, **X** – Xor, **O** – Or.

**16**

Дан текстовый файл. В текстовом файле записана формула следующего вида:

<Формула> ::= <Цифра> | **M**(<Формула>,<Формула>) | **N**(Формула>,<Формула>)

< Цифра > ::= 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9

где буквами обозначены функции:

**M** – определение максимума, **N** – определение минимума.

Используя ***стек***, вычислить значение заданного выражения.

**17**

Дан текстовый файл. Используя ***стек***, проверить, является ли содержимое текстового файла правильной записью формулы вида:

< Формула > ::= < Терм > | < Терм > + < Формула > | < Терм > - < Формула >

< Терм > ::= < Имя > | (< Формула >)

< Имя > ::= **x** | **y** | **z**

**18**

В текстовом файле хранится выражение, записанное в постфиксной форме. Используя ***стек,*** вычислить значение выражения.

*Пример выражения:* **2 3 5 + * 7 6 - * => 16**

**19**

В текстовом файле хранится выражение, записанное в инфиксной форме. Используя ***стек***, перевести его в постфиксную форму и в таком виде записать в новый текстовый файл.

*Пример выражения:* **a + b / c / d * e => a b c / d / e * +**

**20**

В текстовом файле хранится выражение, записанное в постфиксной форме. Используя ***стек***, перевести его в инфиксную форму и в таком виде записать в новый текстовый файл.

*Пример выражения:* ***a b + c * d – f * => ((a + b) * c – d) * f***

## Лабораторная работа №13
## Классы. Объекты

***Цель лабораторной работы:*** *изучение структуры, свойств и видов объектов; изучение способов доступа к полям и правил вызова методов объектов; получение навыков объектно-ориентированного программирования на языке С/С++.*

***Задание на программирование:*** *используя технологию объектно-ориентированного программирования разработать два варианта программы, реализующей движущийся графический объект в соответствии с индивидуальным заданием:*

*- с использованием статического объекта;*

*- с использованием динамического объекта.*

***Порядок выполнения работы:***

1) Получить у преподавателя индивидуальное задание.

2) Разработать иерархию и структуру объектов, связанных на принципах наследования, в соответствии с индивидуальным заданием. Дерево наследования должно содержать не менее трех уровней.

3) Описать типы объектов и методы обработки их полей.

4) Составить две программы на языке С/С++, реализующие движение графического объекта по заданной траектории: в виде динамического объекта и в виде статического объекта описанного типа.

5) Проверить и продемонстрировать преподавателю работу программ.

6) Оформить *отчет о лабораторной работе* в составе: *постановка задачи, математическая модель, схема алгоритма решения, спецификация функций, тексты программ, контрольные примеры.*

## Варианты индивидуальных заданий

**1**
Движение закрашенного прямоугольника по прямоугольному контуру.

**2**
Движение окружности по окружности.

**3**
Движение закрашенного квадрата по окружности.

**4**
Движение треугольника по треугольному контуру.

**5**
Движение закрашенного эллипса по эллиптическому контуру.

**6**
Движение закрашенного прямоугольника по треугольному контуру с изменением цвета при изменении направления движения.

**7**
Движение закрашенного треугольника по эллиптическому контуру.

**8**
Движение закрашенного полукруга по полуокружности.

**9**
Движение закрашенного круга по кромке экрана с изменением цвета при изменении направления движения.

**10**
Движение закрашенного полукруга по кромке экрана с поворотом на 90 градусов в углах экрана.

**11**
Движение отрезка линии в центре экрана по вертикали сверху вниз и обратно с изменением цвета.

**12**
Движение отрезка линии по диагонали экрана из левого нижнего угла в правый верхний угол и обратно с изменением цвета.

**13**
Движение закрашенного прямоугольника по синусоиде по середине экрана.

**14**
Движение закрашенного треугольника в центре экрана по синусоиде сверху вниз.

**15**
Движение закрашенного круга по синусоиде из левого нижнего угла экрана в правый верхний угол.

**16**
Движение закрашенного квадрата по синусоиде из левого верхнего угла экрана в правый нижний угол с изменением цвета.

**17**
Движение креста из двух отрезков линии по синусоиде по середине экрана слева направо и обратно.

**18**
Движение цветного сектора по синусоиде по середине экрана справа налево и обратно.

**19**
Движение треугольника экрана по синусоиде по середине экрана справа налево и обратно.

**20**
Движение окружности по треугольному контуру с изменением цвета при изменении направления движения.

**21**
Движение закрашенного прямоугольника по полуокружности.

**22**
Движение закрашенного полукруга по треугольному контуру.

**23**
Движение окружности по синусоиде по середине экрана справа налево и обратно.

**24**
Движение закрашенного круга по треугольному контуру.

## *Примеры программ*

```
//Движение прямоугольника по треугольному контуру.
//Динамические объекты.
#include <conio.h>
#include <graphics.h>
#include <iostream.h>
#include <process.h>
#include <string.h>
#include <dos.h>

class graphworld
  {int driver,mode,grerror,colb,bkcl;
   char path[80];
   public:
     graphworld();
     void closegraphworld();
  };

graphworld::graphworld()
  {strcpy(path,"c:\\turboc30\\bgi");
   driver=0;
   initgraph(&driver,&mode,path);
   grerror=graphresult();
   if(grerror!=grOk)
     {cout<<"\nОшибка открытия графического режима";
      abort;
     }
   setcolor(RED);
   setbkcolor(BLACK);
   cleardevice();
  }

void graphworld::closegraphworld()
     {cleardevice();
      closegraph();
     }

class location
   {protected:
      int x,y;
    public:
      location(int initx,int inity);
      int getx();
      int gety();
   };

location::location(int initx, int inity)
  {x=initx;
   y=inity;
  }

int location::getx()
  {return x;
  }


int location::gety()
  {return y;
  }
```

```
class point:public location
  {protected:
     int visible;
     void setvisible(int pr);
   public:
     point(int initx,int inity);
     ~point();
     virtual void show();
     virtual void hide();
     int getvisible();
     void moveto(int nx,int ny);
   };

point::point(int initx,int inity):location(initx,inity)
       {
       }

point::~point()
       {hide();
       }

void point::moveto(int nx,int ny)
     {hide();
      x=nx;
      y=ny;
      show();
     }

void point::setvisible(int pr)
     {visible=pr;
     }

void point::show()
     {putpixel(x,y,getcolor());
      setvisible(1);
     }

void point::hide()
     {putpixel(x,y,getbkcolor());
      setvisible(0);
     }

class pramoug:public point
   {int dx,dy;
    public:
      pramoug(int initx,int inity,int initdx,int initdy);
     ~pramoug();
     void show();
     void hide();
   };

pramoug::pramoug(int initx,int inity,int initdx,int initdy): point(initx,inity)
     {dx=initdx;
      dy=initdy;
     }

void pramoug::show()
     {line(x,y,x,y+dy);
      line(x,y+dy,x+dx,y+dy);
      line(x+dx,y+dy,x+dx,y);
      line(x,y,x+dx,y);
     }
```

```
void pramoug::hide()
     {int r;
      r=getcolor();
      setcolor(getbkcolor());
      line(x,y,x,y+dy);
      line(x,y+dy,x+dx,y+dy);
      line(x+dx,y+dy,x+dx,y);
      line(x,y,x+dx,y);
      setcolor(r);
     }

pramoug::~pramoug()
     {hide();
     }

void main(void)
{graphworld world;
 pramoug *pt;
 int x,y;
 getch();
 cleardevice();
 x = 150;
 y = 100;
 pt = new pramoug(x,y,200,100);
 delay(750);
 pt -> show();
 do
   {do
      {x += 3; y++;
       pt -> moveto(x,y);
       delay(5);
      }
    while(!(y >= 200));
    do
      {x--; y++;
       pt -> moveto(x,y);
       delay(5);
      }
    while(!(y >= 400));
    do
      {x--; y -= 3;
       pt -> moveto(x,y);
       delay(5);
      }
    while(!(y <= 100));
   }
 while(!(kbhit()));
 delete pt;
 getch();
 world.closegraphworld();
}
```

```
//Движение прямоугольника по треугольному контуру.
//Статические объекты.
#include <conio.h>
#include <graphics.h>
#include <iostream.h>
#include <process.h>
#include <string.h>
#include <dos.h>

class graphworld
  {int driver,mode,grerror,colb,bkcl;
   char path[80];
   public:
     graphworld();
     void closegraphworld();
  };

graphworld::graphworld()
  {strcpy(path,"c:\\turboc30\\bgi");
   driver=0;
   initgraph(&driver,&mode,path);
   grerror=graphresult();
   if(grerror!=grOk)
     {cout<<"\nОшибка открытия графического режима";
      abort;
     }
   setcolor(RED);
   setbkcolor(BLACK);
   cleardevice();
  }

void graphworld::closegraphworld()
     {cleardevice();
      closegraph();
     }

class location
   {protected:
      int x,y;
    public:
      location(int initx,int inity);
      int getx();
      int gety();
   };

location::location(int initx, int inity)
  {x=initx;
   y=inity;
  }

int location::getx()
  {return x;
  }

int location::gety()
  {return y;
  }

class point:public location
  {protected:
     int visible;
     void setvisible(int pr);
   public:
```

```
     point(int initx,int inity);
     ~point();
     virtual void show();
     virtual void hide();
     int getvisible();
     void moveto(int nx,int ny);
   };

point::point(int initx,int inity):location(initx,inity)
       {
       }

point::~point()
       {hide();
       }

void point::moveto(int nx,int ny)
     {hide();
      x=nx;
      y=ny;
      show();
     }

void point::setvisible(int pr)
     {visible=pr;
     }

void point::show()
     {putpixel(x,y,getcolor());
      setvisible(1);
     }

void point::hide()
     {putpixel(x,y,getbkcolor());
      setvisible(0);

     }

class pramoug:public point
   {int dx,dy;
    public:
      pramoug(int initx,int inity,int initdx,int initdy);
     ~pramoug();
     void show();
     void hide();
   };

pramoug::pramoug(int initx,int inity,int initdx,int initdy): point(initx,inity)
     {dx=initdx;
      dy=initdy;
     }

void pramoug::show()
     {line(x,y,x,y+dy);
      line(x,y+dy,x+dx,y+dy);
      line(x+dx,y+dy,x+dx,y);
      line(x,y,x+dx,y);
     }

void pramoug::hide()
     {int r;
      r=getcolor();
      setcolor(getbkcolor());
```

```
       line(x,y,x,y+dy);
       line(x,y+dy,x+dx,y+dy);
       line(x+dx,y+dy,x+dx,y);
       line(x,y,x+dx,y);
       setcolor(r);
      }

pramoug::~pramoug()
      {hide();
      }

void main(void)
{graphworld world;
 int x = 150,
     y = 100;
 getch();
 cleardevice();
 pramoug pt(x,y,200,100);
 delay(750);
 pt.show();
 do
   {do
      {x += 3; y++;
       pt.moveto(x,y);
       delay(5);
      }
    while(!(y >= 200));
    do
      {x--; y++;
       pt.moveto(x,y);
       delay(5);
      }
    while(!(y >= 400));
    do
      {x--; y -= 3;
       pt.moveto(x,y);
       delay(5);
      }
    while(!(y <= 100));
   }
 while(!(kbhit()));
 getch();
 world.closegraphworld();
}
```

## Литература

1 Павловская Т.А. С/С++. Программирование на языке высокого уровня. СПб.: Питер, 2001. – 464 с.

2 Павловская Т.А., Щупак Ю.А. С/С++. Структурное программирование: Практикум. СПб.: Питер, 2002. - 240 с.

3 Культин Н.Б. С/С++ в задачах и примерах. СПб.: БХВ-Петербург, 2001. – 288 с.

4 Керниган Б., Ритчи Д., Фьюэр А. Язык программирования Си. М.: Финансы и статистика, 1985. – 384 с.

5 Бондарев В.М., Рублинецкий В.И., Качко Е.Г. Основы программирования. Харьков: Фолио, 1997 г. – 368с.

6 ГОСТ 19.701-90 ЕСПД Схемы алгоритмов, программ, данных и систем. Условные обозначения и правила выполнения.

7 ГОСТ 7.32-2001 Система стандартов по информации, библиотечному и издательскому делу. Отчет по научно-исследовательской работе. Структура и правила оформления.