МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Методические указания

по выполнению лабораторных работ
по дисциплине «Объектно-ориентированное программирование»
для студентов
специальности 10.05.03 Информационная безопасность
автоматизированных систем
специализация Анализ безопасности информационных систем

Ставрополь

Содержание

ВВЕДЕНИЕ

Цель и задачи освоения дисциплины

Целью освоения дисциплины «объектно-ориентированное программирование» является формирование у будущего специалиста по специальности 10.05.03 Информационная безопасность автоматизированных систем специализация «Анализ безопасности информационных систем» заключается в формировании у будущих специалистов навыков разработки, анализа и защиты программного обеспечения с использованием парадигмы ООП, что необходимо для обеспечения информационной безопасности автоматизированных систем.

Задачи дисциплины:

- Разработка безопасного объектно-ориентированного кода
- Анализ защищенности ООП-приложений
- Применение ООП в задачах информационной безопасности
- Работа с фреймворками и библиотеками

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

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++.

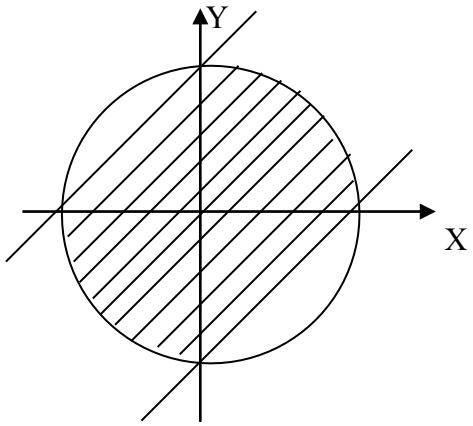
5) *Входные данные* вещественного типа ***float*** вводить с клавиатуры по запросу.
Выходные данные (сообщения) выводить на экран в развернутой форме.

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

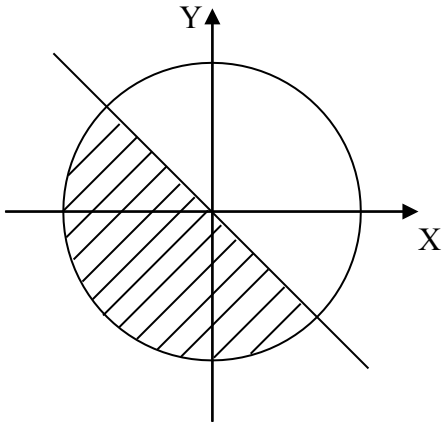
7) Оформить *отчет о лабораторной работе* в составе: *постановка задачи, математическая модель, схема алгоритма решения, текст программы, контрольные примеры.*

Варианты индивидуальных заданий

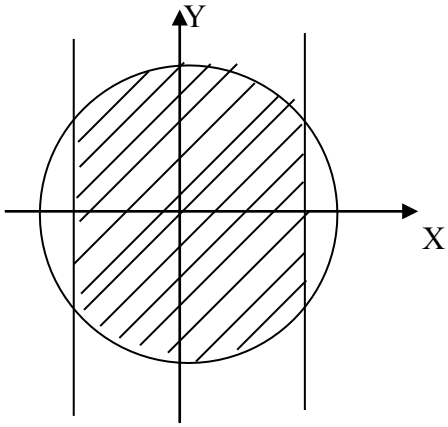
1)



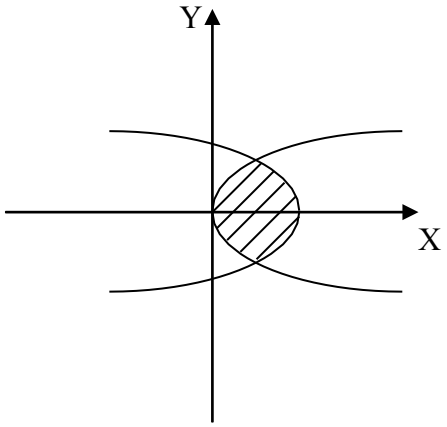
2)



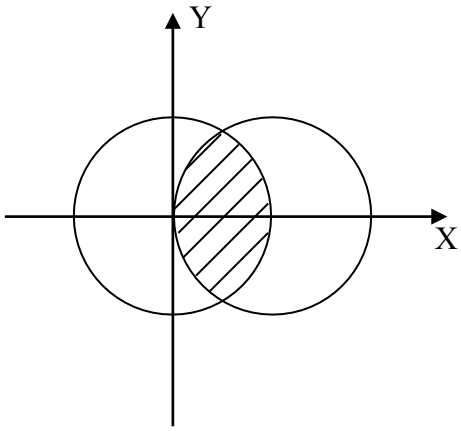
3)



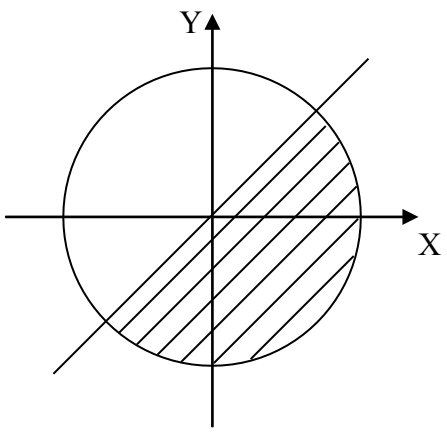
4)



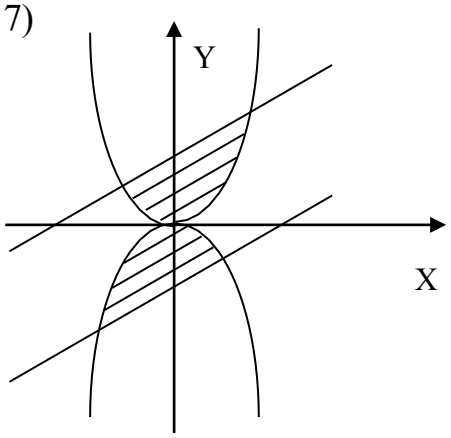
5)



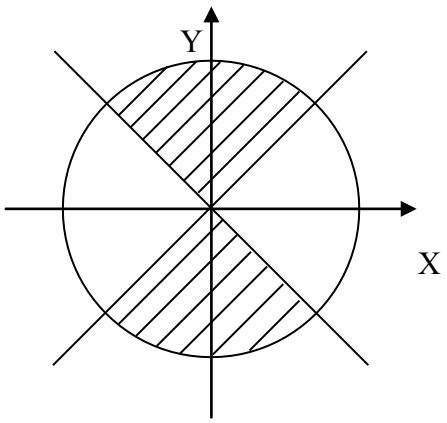
6)



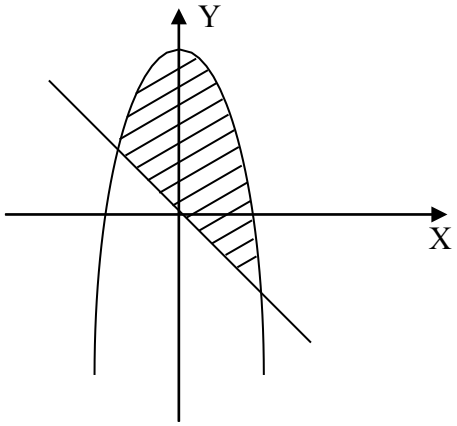
7)



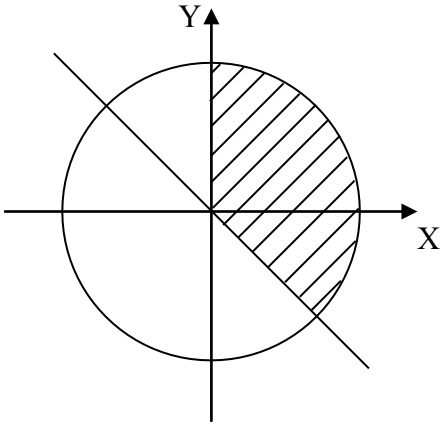
8)



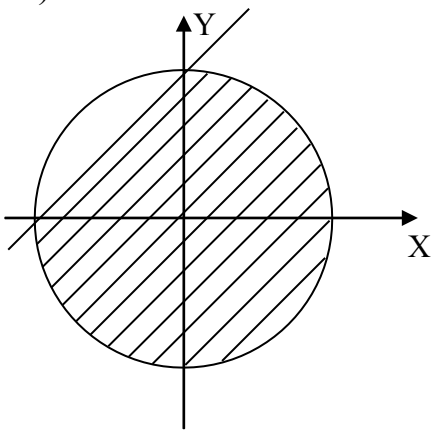
9)



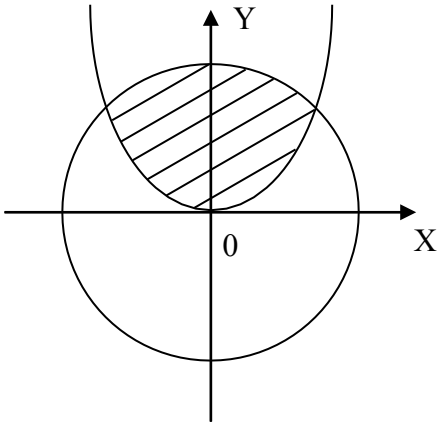
10)



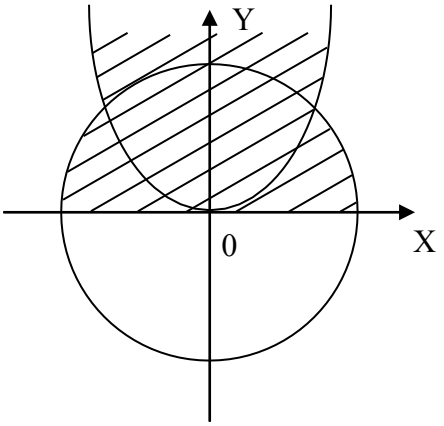
11)



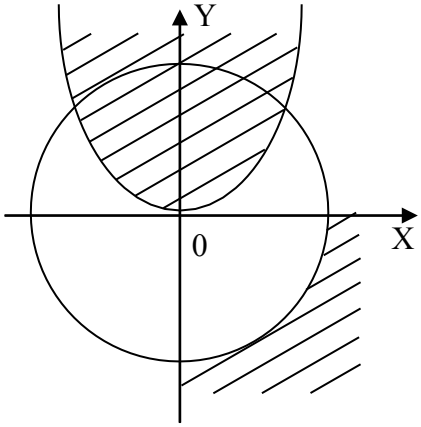
12)



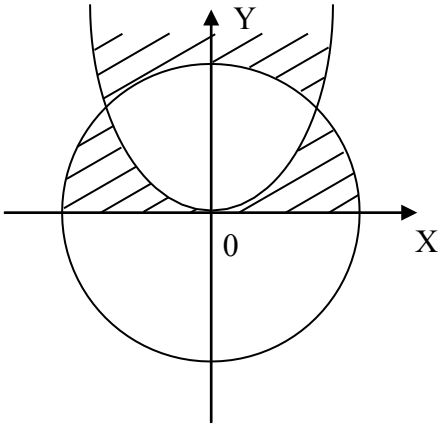
13)



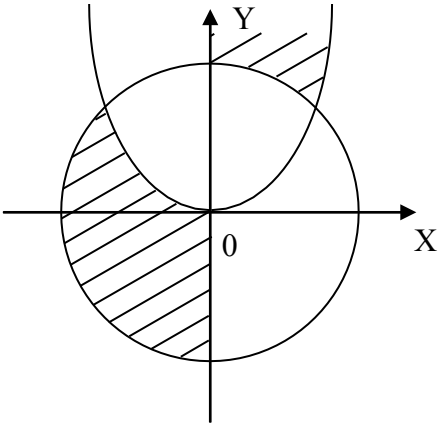
14)



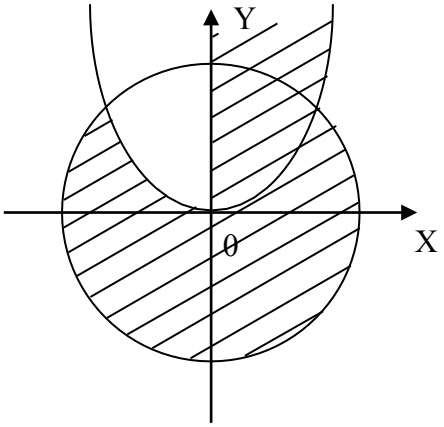
15)



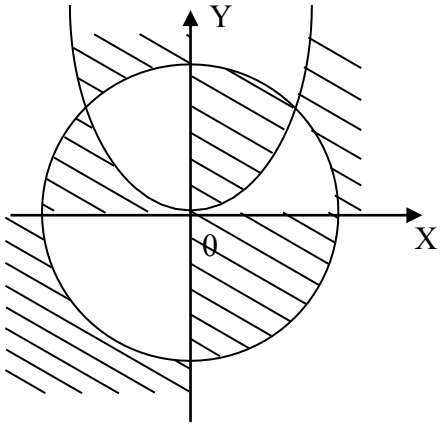
16)



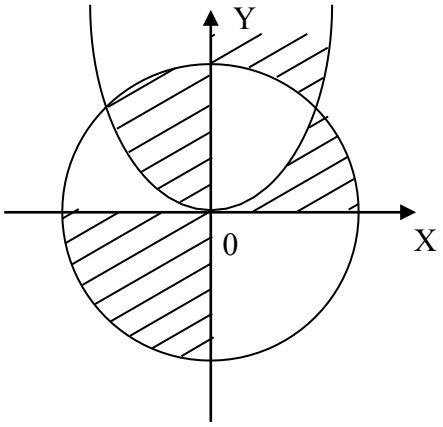
17)



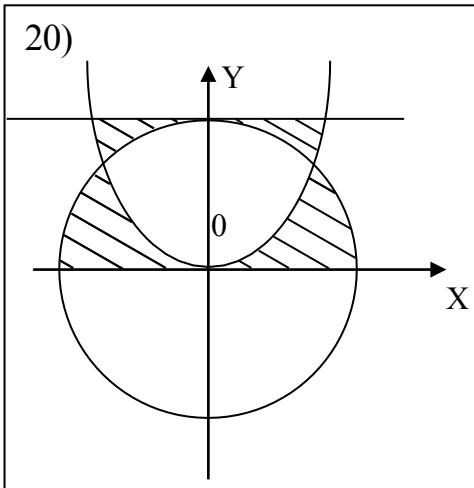
18)



19)

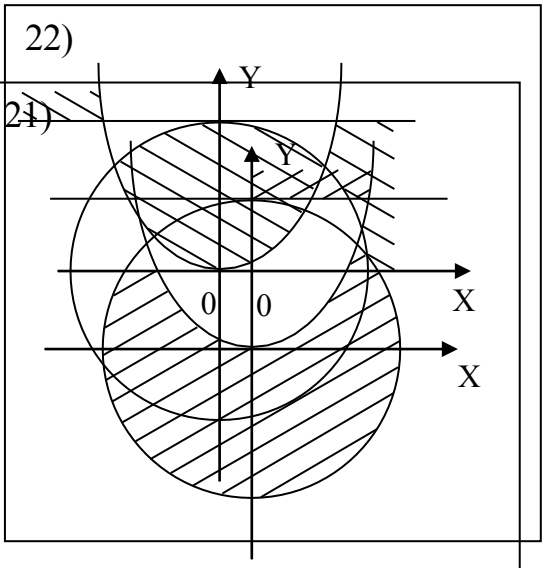


20)

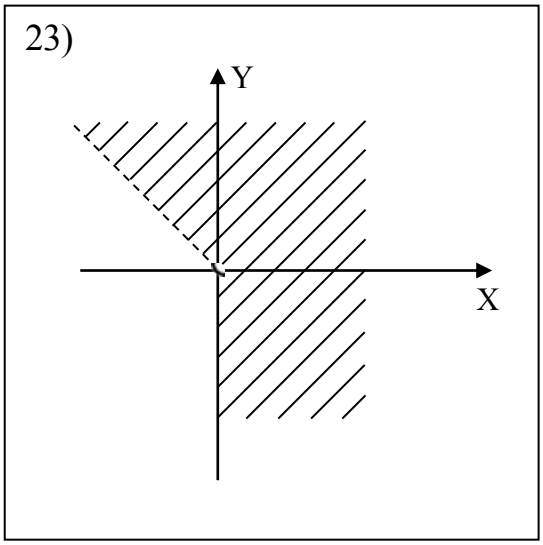


21)

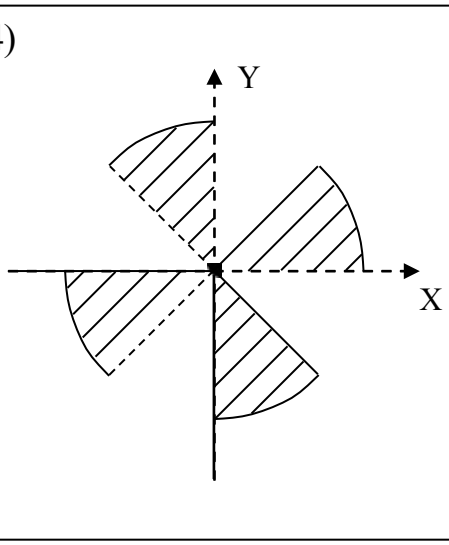
22)



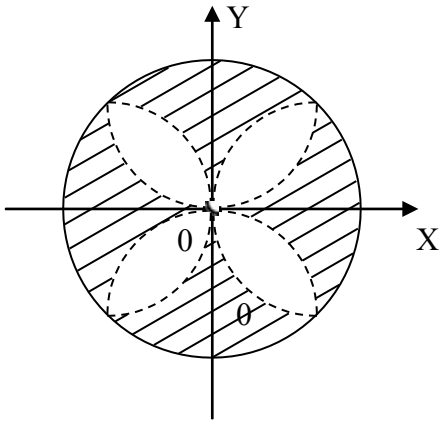
23)



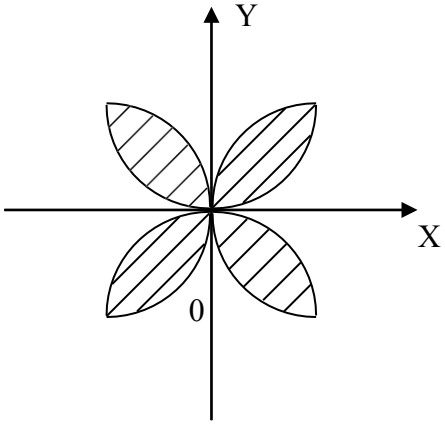
24)



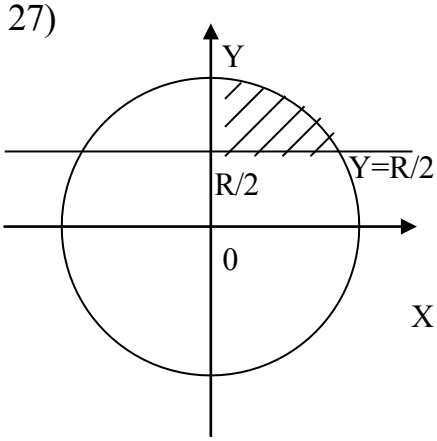
25)



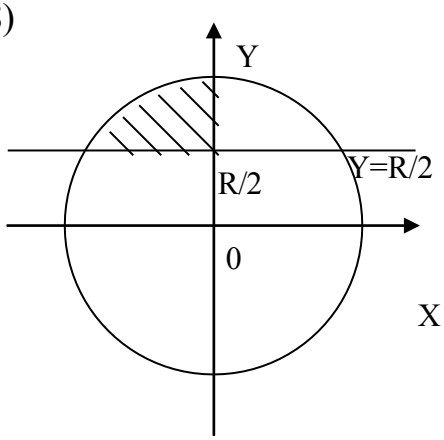
26)



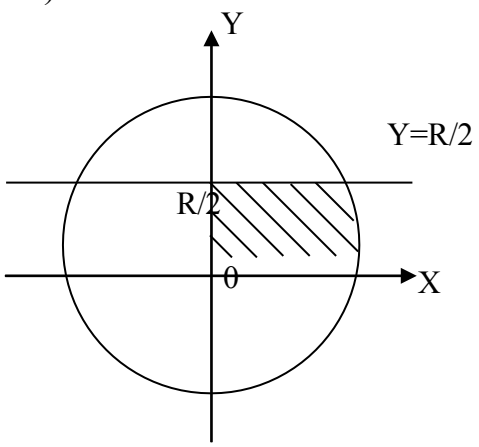
27)



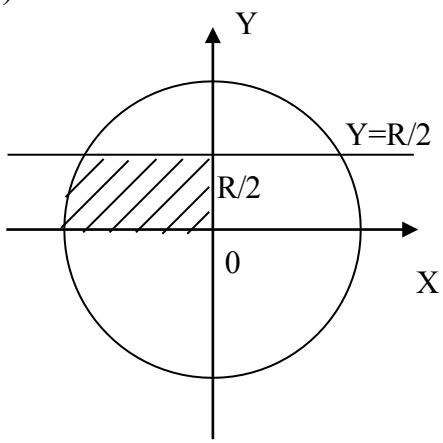
28)



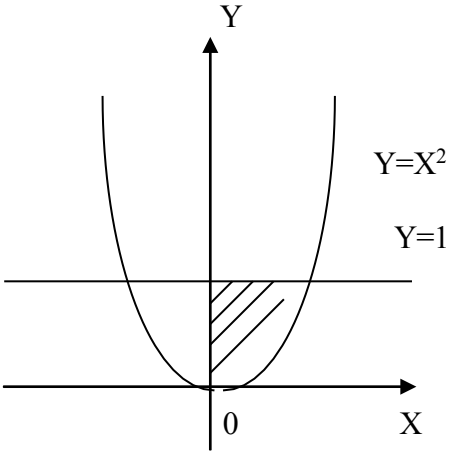
29)



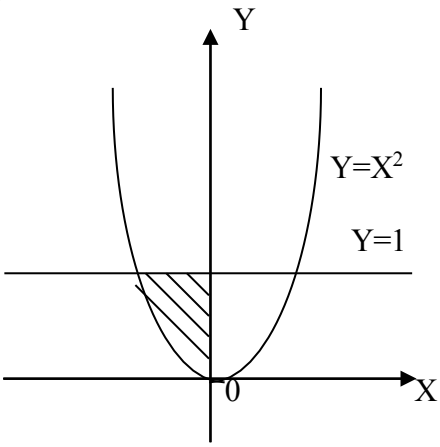
30)



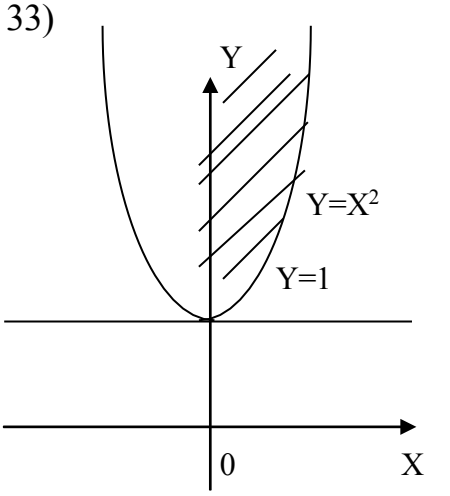
31)



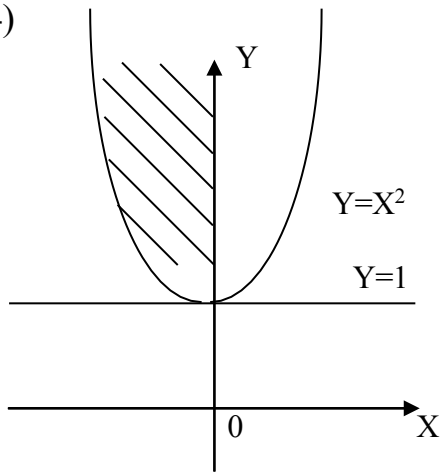
32)



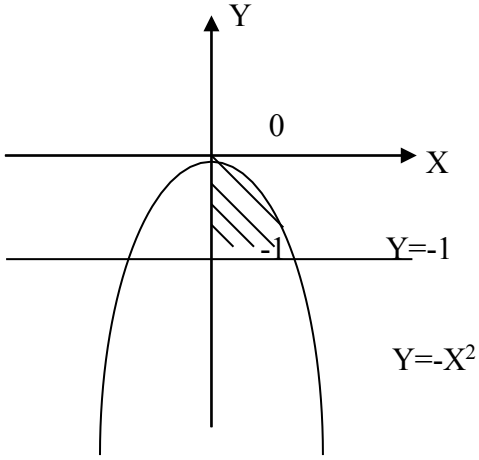
33)



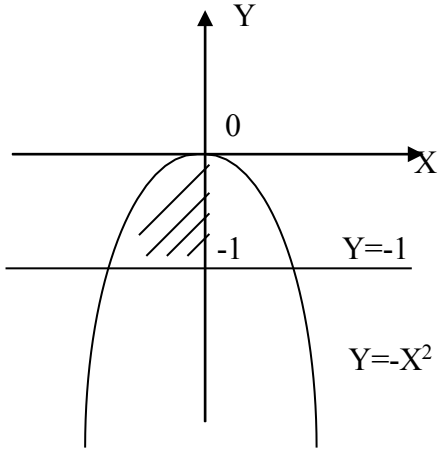
34)



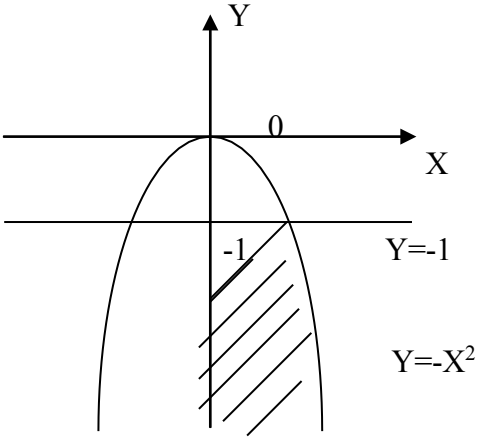
35)



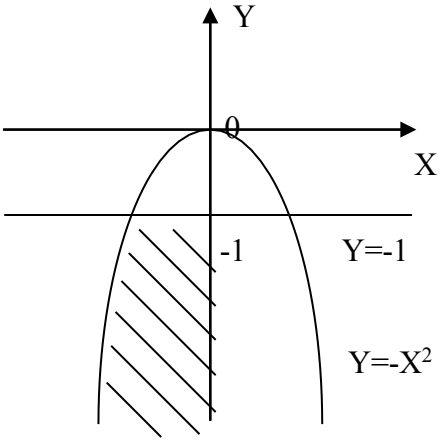
36)



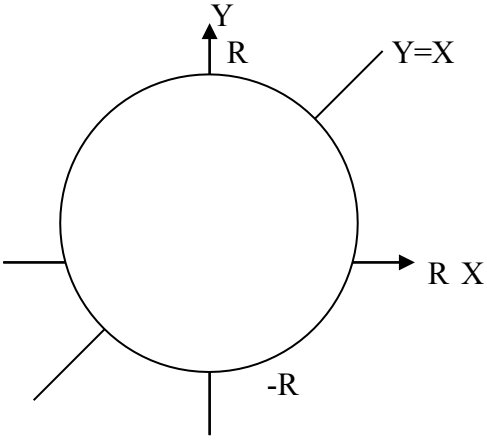
37)



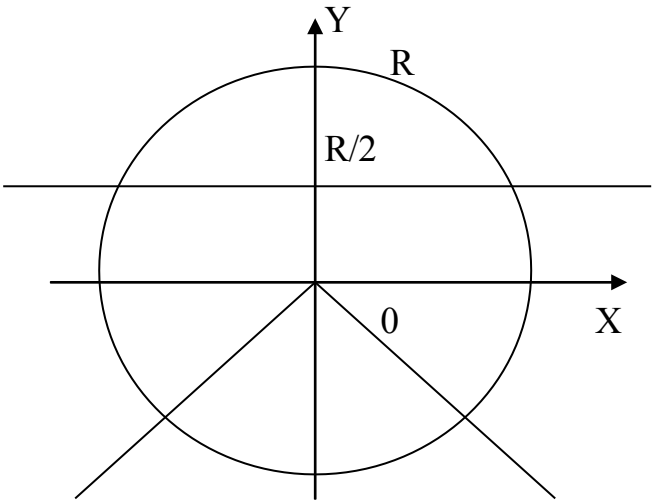
38)

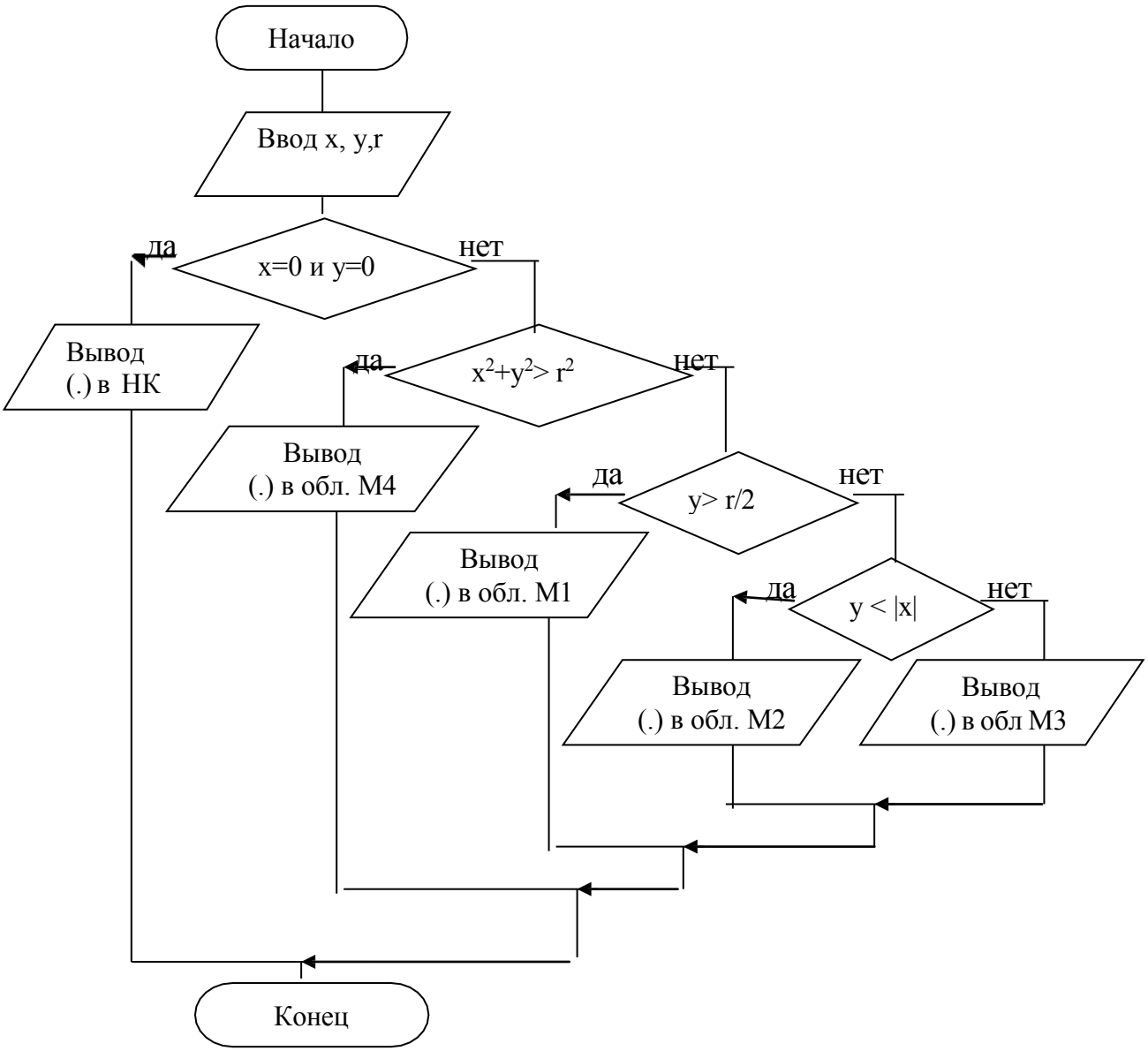


39)



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 C/C++ 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 C/C++.

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:

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

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:

$$P = \prod_{i=1}^{m} X^{X}$$

$$P \bullet \circ (m \tilde{} \quad)$$
$$i \qquad m\ i\ 1$$

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:

$$S \bullet A \sum_{i \bullet m}^{n} (X_i \frac{B}{})^2$$

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

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

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

4

1. Вычислить предел последовательности $\{Y_n\}$ при $n \to \infty$, где $Y_0 = 0$, а Y_n вычисляется по формуле

$$Y_n \bullet \frac{1}{1\ Y_{n1}}; n \bullet 1,2,3...$$

Значение Y_0 вводится с клавиатуры. Вычисления прекращаются при выполнении условия $|Y_n - Y_{n-1}| < \varepsilon$ **5**

1. По введенному с клавиатуры значению X вычислить S:

$$S \bullet \frac{(\tilde{X}\ 2) (\tilde{X}\ 4) (\tilde{X}\ 8) \ldots (\tilde{X}\ 128)}{(\tilde{X}\ 1) (\tilde{X}\ 3) (\tilde{X}\ 7) \ldots (\tilde{X}\ 127)}$$

2. Вычислить предел последовательности $\{Y_n\}$ при $n \to \infty$, где Y_n вычисляется по формулеЗначения Y_0, Y_1 вводятся с клавиатуры. Вычисления прекратить при выполнении условия $|Y_n - Y_{n-1}| < \varepsilon$.

6

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

$$Y_n \bullet \frac{Y_{\tilde{n}2}.\ 0.5 \ Y_{\tilde{n}3}; n}{Y \quad 2Y} \bullet 3,4,5...$$
$$\quad {}^{n}_{2} \qquad {}^{4}_{\tilde{n}3} 1.5$$
$$\quad {}^{-n}_{2}$$

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

7

1. Для заданного с клавиатуры значения N найти (2*N+1)!! по формуле
$$(2*N+1)!! = 1*3*5*\ldots*(2*N-1)*(2*N+1).$$

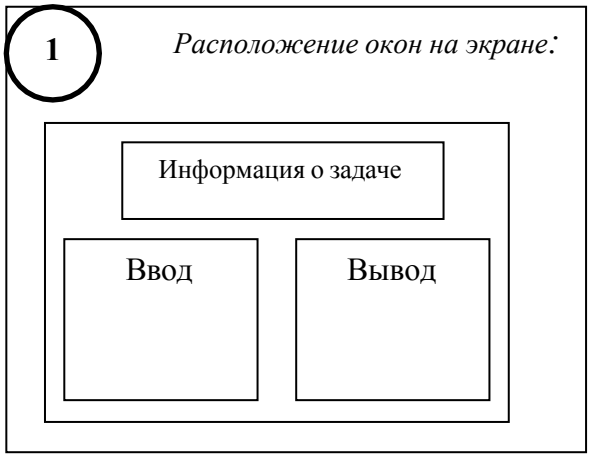
2. Последовательность функций $Y_n = Y_n(x)$, где $0 \le x \le 1$ определяется следующим образом:

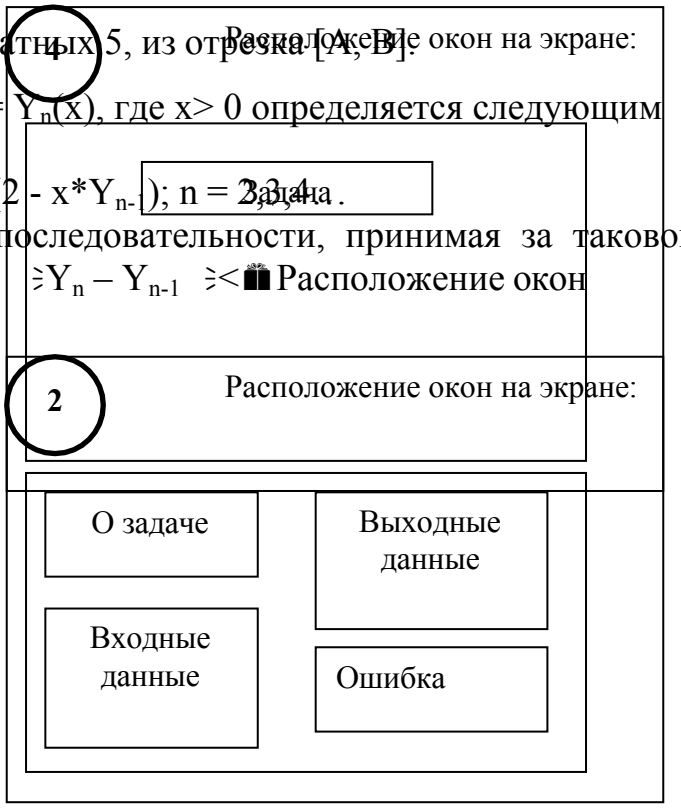
3. При заданном x найти предел последовательности, принимая за таковой значение Y_n, удовлетворяющее условию $|Y_n - Y_{n-1}| < \varepsilon$.

8

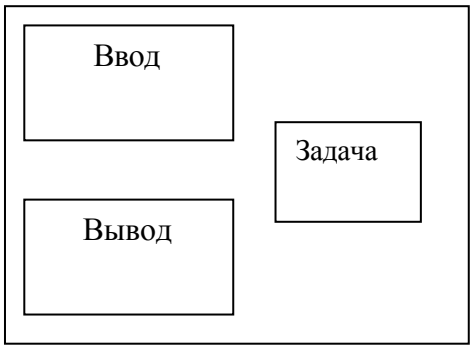
1. Найти сумму всех целых чисел, кратных 5, из отрезка [A; B].

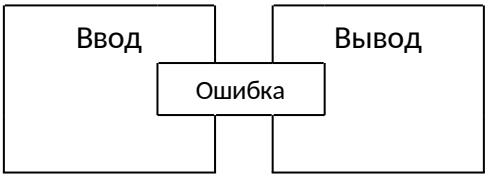
2. Последовательность функций $Y_n = Y_n(x)$, где $x > 0$ определяется следующим образом:
$$Y_1 = x;\ Y_n = Y_{n-1}*(2 - x*Y_{n-1});\ n = 2,3,4\ldots$$
При заданном X найти предел последовательности, принимая за таковой значение Y_n, удовлетворяющее условию $|Y_n - Y_{n-1}| < \varepsilon$



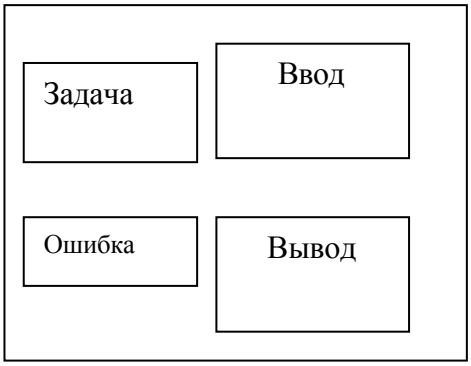


3 Расположение окон на экране:

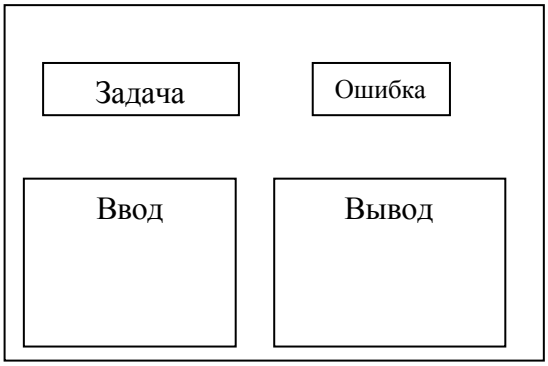




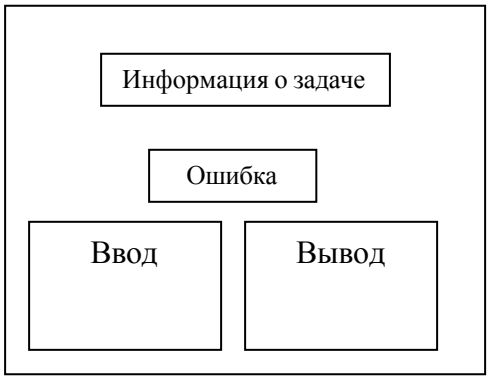
5 Расположение окон на экране:



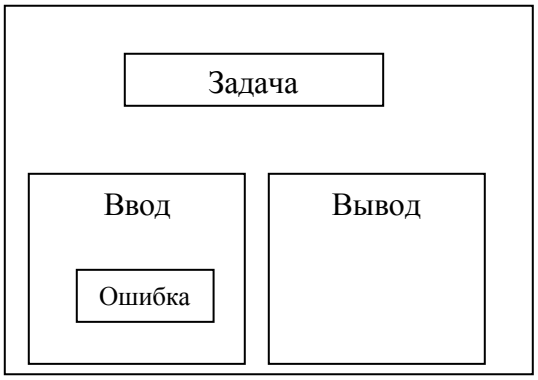
6 Расположение окон на экране:



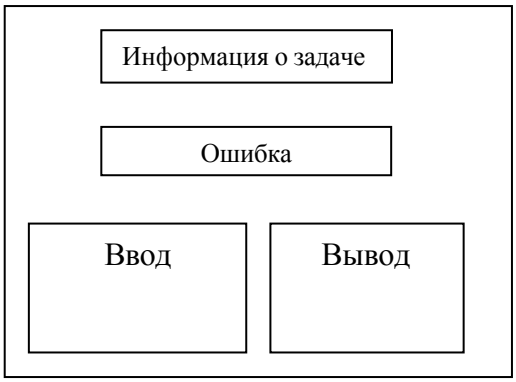
7 Расположение окон на экране:



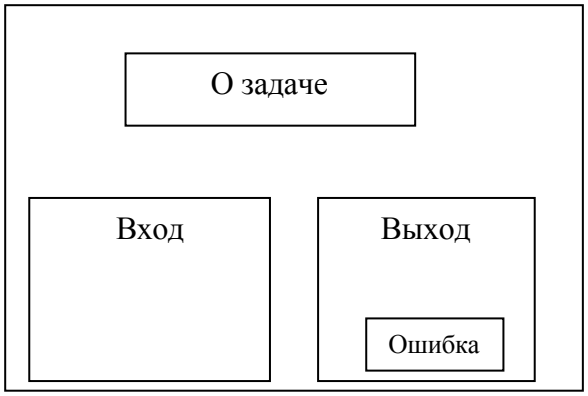
8 Расположение окон на экране:



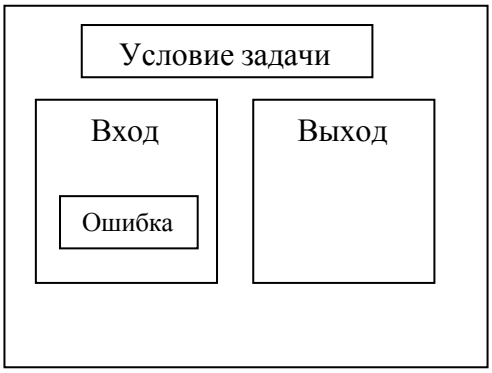
9 Расположение окон на экране:



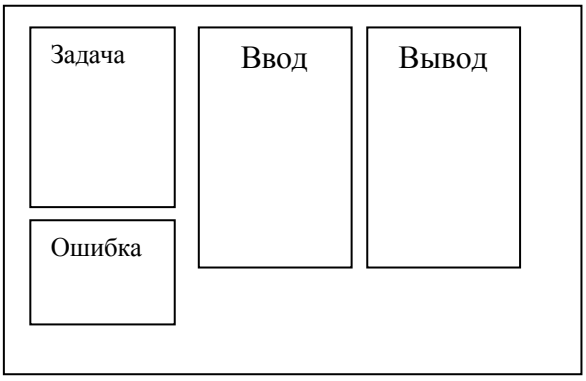
10 Расположение окон на экране:



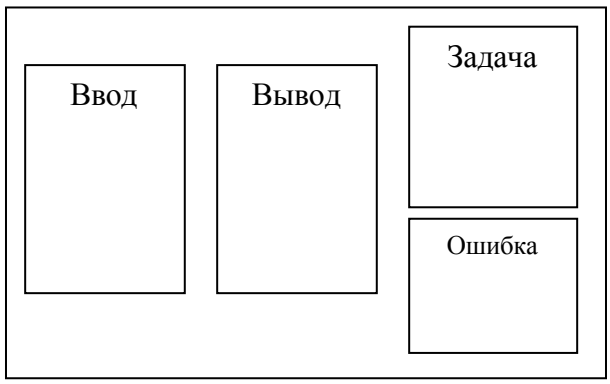
11 Расположение окон на экране:



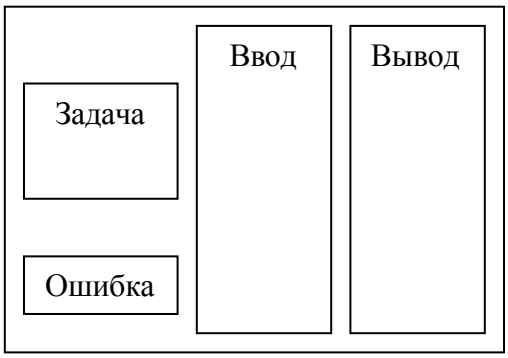
12 Расположение окон на экране:



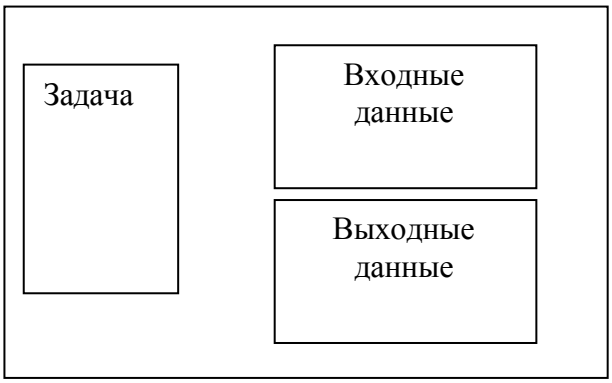
13 Расположение окон на экране:



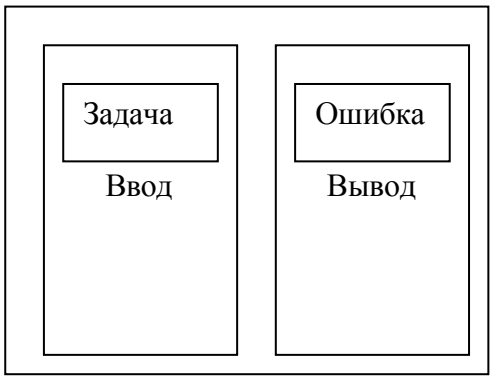
14 Расположение окон на экране:



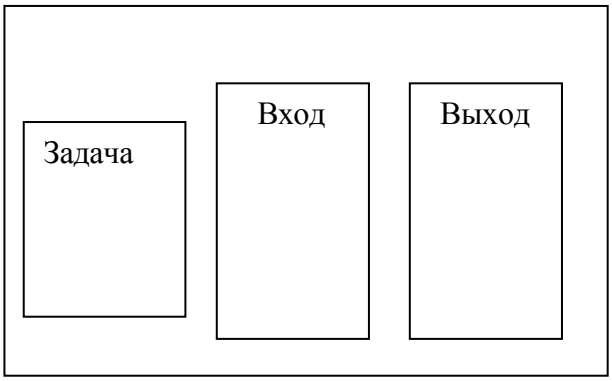
15 Расположение окон на экране:



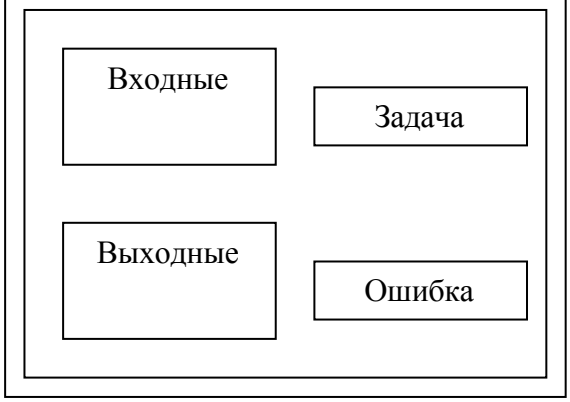
16 Расположение окон на экране:



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 С/С++.

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.

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,…, 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, …, 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,…, 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, …, 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 K, то первую по порядку компоненту этого массива, большую всех остальных компонент, заменить значением K.

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 массивn $b_1,b_2,\ldots,b_n$, 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.

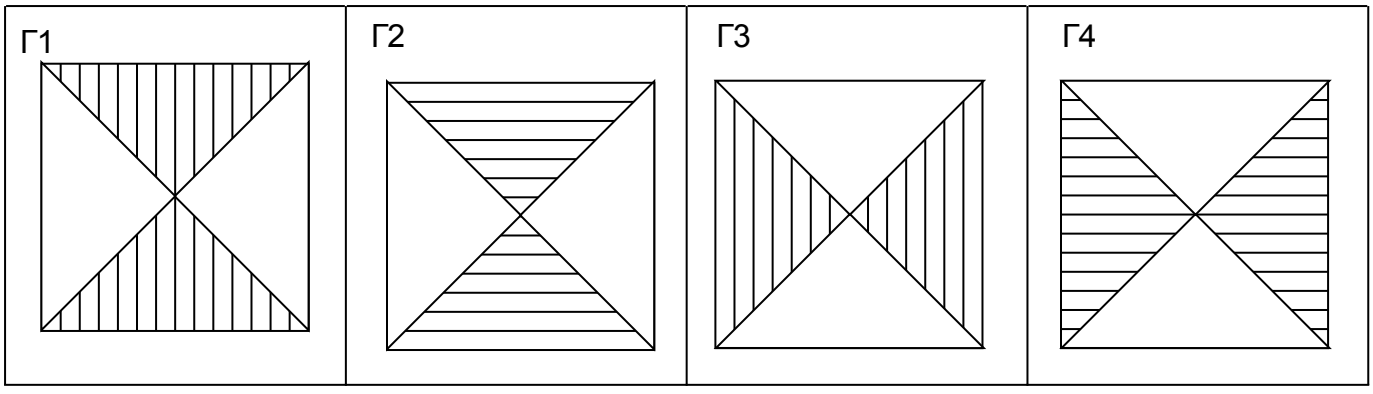
9
Сортировка по возрастанию методом вставки.

10
Сортировка по убыванию методом вставки.

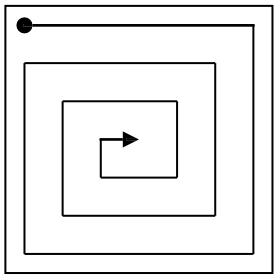
11
Быстрая сортировка по возрастанию.

12
Быстрая сортировка по убыванию.

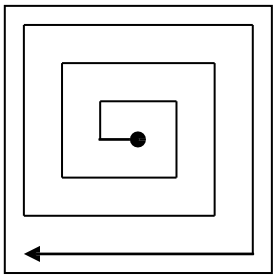
Сортируемые фрагменты матриц



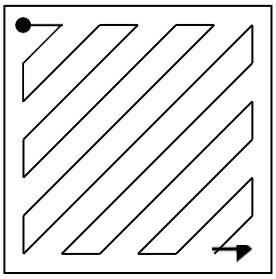
Г5



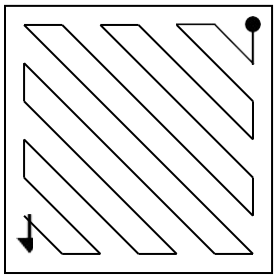
Г6



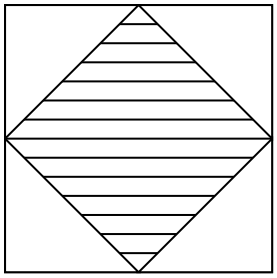
Г7



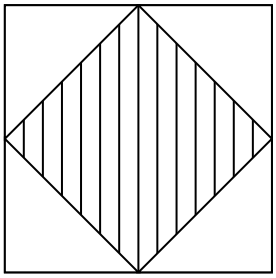
Г8



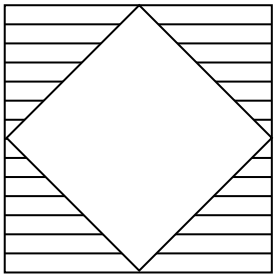
В1



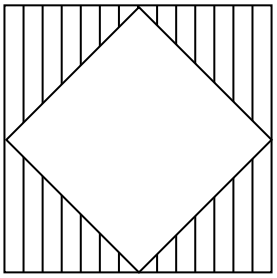
В2



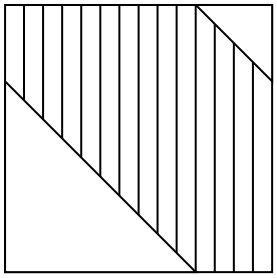
В3



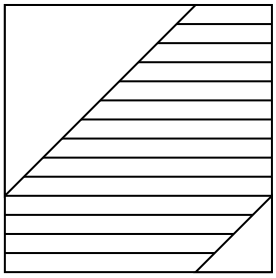
В4



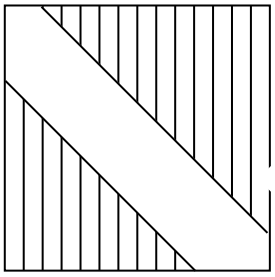
В5



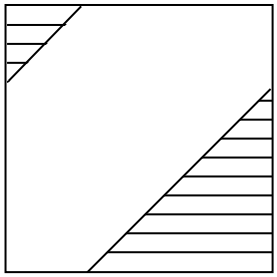
В6



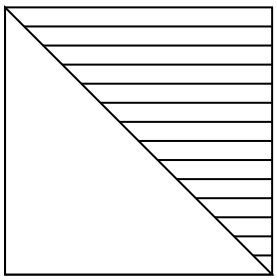
В7



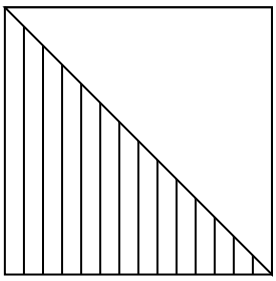
В8



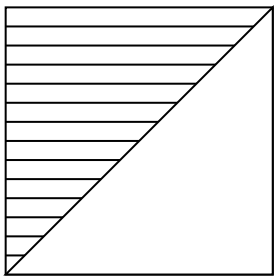
Б1



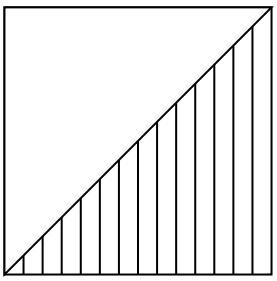
Б2



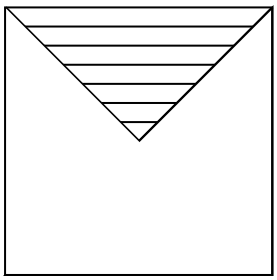
Б3



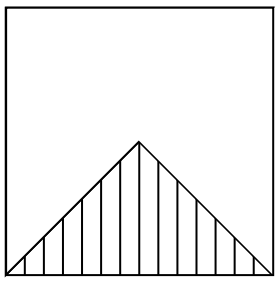
Б4



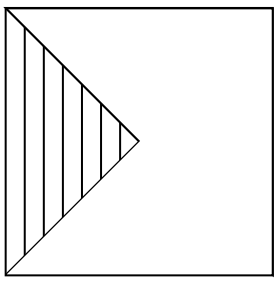
Б5



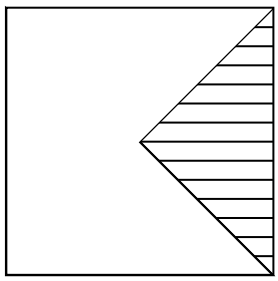
Б6



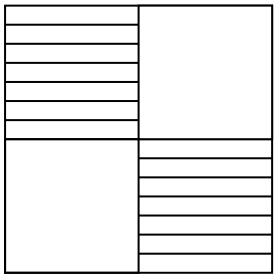
Б7



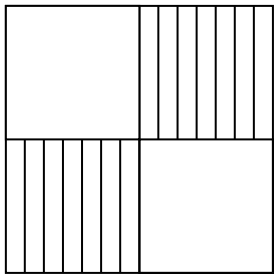
Б8



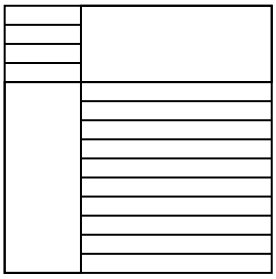
А1



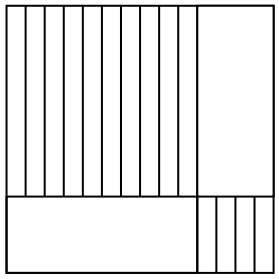
А2



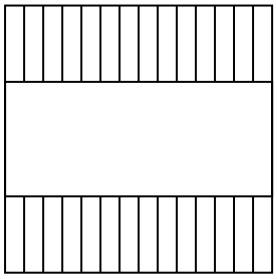
А3



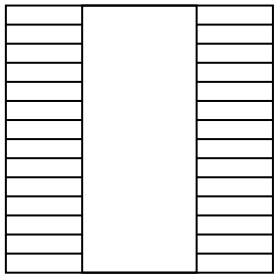
А4



А5

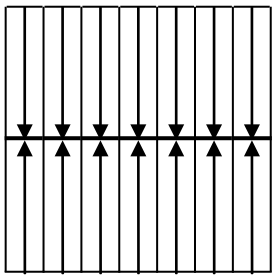


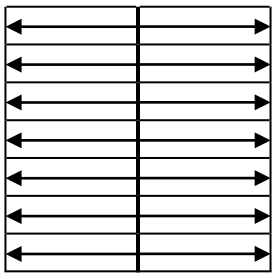
А6



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++;
   }
}
```

Список литературы

Перечень основной литературы:

1. Страуструп, Б. Программирование: принципы и практика с использованием C++ / Б. Страуструп. — М.: Вильямс, 2020. — 1328 с. — ISBN 978-5-8459-2089-3.

2. Лутц, М. Изучаем Python / М. Лутц. — СПб.: Питер, 2022. — 1648 с. — ISBN 978-5-4461-1599-3.

3. Орлов, С. А. Программирование на C++ в защищённых системах / С. А. Орлов, Б. Я. Цилькер. — СПб.: Питер, 2020. — 352 с. — ISBN 978-5-4461-1345-6.

4. Дейтел, П. Java. Руководство для начинающих / П. Дейтел, Х. Дейтел. — М.: Вильямс, 2021. — 816 с. — ISBN 978-5-8459-2101-2.

Перечень дополнительной литературы:

1. Блинов, И. Н. Java. Объектно-ориентированное программирование / И. Н. Блинов, В. С. Романчик. — Минск: Четыре четверти, 2018. — 560 с. — ISBN 978-985-581-123-4.

2. Гамма, Э. Приемы объектно-ориентированного проектирования. Паттерны проектирования / Э. Гамма, Р. Хелм, Р. Джонсон, Дж. Влиссидес. — СПб.: Питер, 2019. — 496 с. — ISBN 978-5-4461-1199-5.

3. Макконнелл, С. Совершенный код / С. Макконнелл. — М.: БХВ-Петербург, 2021. — 896 с. — ISBN 978-5-9775-3721-1.

4. Кузнецов, М. В. Безопасное программирование на Python / М. В. Кузнецов, И. В. Симдянов. — СПб.: БХВ-Петербург, 2022. — 400 с. — ISBN 978-5-9775-0876-1.

5. Howard, M. Writing Secure Code / M. Howard, D. LeBlanc. — Microsoft Press, 2021. — 768 p. — ISBN 978-0-7356-8852-6.

6. Вендров, А. М. Проектирование программного обеспечения. Учебник / А. М. Вендров. — М.: Финансы и статистика, 2020. — 512 с. — ISBN 978-5-279-03456-7.

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение
высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Методические указания

по выполнению курсового проекта по дисциплине

«**Объектно-ориентированное программирование**»

для студентов специальности

10.05.03 Информационная безопасность автоматизированных систем

Введение

Настоящие рекомендации предназначены для студентов всех форм обучения при выполнении курсовых работ, для руководителей курсовых работ с целью формирования единых требований при разработке и защите работ. Рекомендации содержат предложения по структуре курсовой работы (далее - работа), объему, содержанию, оформлению, процедуре допуска к защите и порядку защиты работ.

Целями выполнения курсовой работы являются:

- обогащение знаний студентов,
- обучение методам теоретического анализа явлений и закономерностей науки,
- выработка навыков применения теоретических знаний к комплексному решению профессиональных задач, использования справочной литературы, методов математической обработки экспериментальных данных, компьютерных технологий.

Системой курсовых работ (проектов) студент подготавливается к выполнению выпускной квалификационной работы.

Перед прочтением Методических указаний ознакомьтесь с основными ошибками, которые допускают студенты, приступая к написанию курсовой работы. Возможно, это мотивирует на более внимательное отношение к требованиям по написанию курсовой работы.

1. Выбрав тему, студент выходит в сеть Интернет и через поисковую систему (Yandex, Google и т.д. – зависит от предпочтений) находит курсовую работу по данной теме, меняет реквизиты, Ф.И.О. автора и сдаёт преподавателю. В результате получает рецензию: «Плагиат. На переработку».
2. Аналогичные действия п.1, но студент подходит к решению проблемы творчески и находит несколько курсовых работ по данной теме. Компонует материал по своему усмотрению и сдает преподавателю под своим авторством. В результате получает рецензию: «Плагиат. На переработку». Почему дана такая рецензия? По трем основным причинам: во-первых, написание курсовых работ – распространённый бизнес в Интернете и качественный материал бесплатно не выкладывают в сети, таким образом, компоновать приходится из «третьесортного сырья»; во-вторых, курсовая уже написана и защищена ранее, то есть она уже устарела и не отражает современных реалий, в-третьих, такие работы редко соответствуют требованиям, предъявляемым кафедрой к курсовым работам.
3. Студент принимает постороннюю помощь в написании работы. Здесь есть несколько аспектов, но следует обратить внимание на два основных. Во-первых, помощь исходит, как правило, от одного физического или юридического лица для группы студентов. В результате преподаватель получает на проверку несколько работ на разные темы, написанные «одной рукой». Формальных причин для отрицательной рецензии у преподавателя нет, особенно если работа выполнена в строгом соответствии с требованиями, но, как показывает практика, защита таких работ проходит безуспешно. Во-вторых, студент, не приложивший усилий к данной работе, как правило, очень плохо в ней ориентируется, особенно, когда преподаватель уточняет аспекты рассматриваемой темы. Но так как Вы уже вышли на защиту, то получаете заслуженную оценку «неудовлетворительно».
4. Вы самостоятельно осуществляете работу, но так же самостоятельно выбираете требования к курсовой работе. Если Вам они нравятся и понятны, то исполняете, а если не нравятся, пропускаете. Это очень распространённая ошибка. Выполнять надо все без исключения требования к курсовой работе. Если в процессе работы возникли вопросы, следует по ним получить консультацию у научного руководителя.

Это краткий перечень основных ошибок, которые недопустимо совершать после прочтения данных Методических указаний.

1. ОРГАНИЗАЦИЯ ВЫПОЛНЕНИЯ И ЗАЩИТЫ КУРСОВОЙ РАБОТЫ

Виды работ по этапам выполнения курсовой работы:

а) Предварительный этап:

- выбор темы курсовой работы и оценка возможности раскрытия данной темы;
- подача заявления на закрепление темы;

б) Основной этап:

- сбор материала;
- оформление курсовой работы;

в) Заключительный этап:

- рецензирование работы руководителем и допуск к защите;
- защита работы.

1.1 Предварительный этап

Темы курсовых работ определяются выпускающей кафедрой. Студенту предоставлено право выбора темы курсовой работы. Задание к выбранной теме выдается руководителем работы (проекта) и регистрируется в кафедральном журнале.

1.2 Основной этап

Составление плана курсовой работы

План курсовой работы представляет собой составленный в определенном порядке перечень глав и развернутый перечень вопросов, которые должны быть освещены. Правильно составленный план работы помогает систематизировать материал, обеспечивает последовательность его изложения. План работы составляется самостоятельно, с учетом замысла и индивидуального подхода. План работы рекомендуется **согласовать с руководителем курсовой работы**. В процессе работы план работы может уточняться.

Подбор литературы

Курсовая работа выполняется на основе глубокого изучения литературных источников. Литература по теме работы может быть подобрана студентом при помощи предметных и алфавитных каталогов библиотек. Для этих целей могут быть использованы каталоги книг, указатели журнальных статей, специальные библиографические справочники, тематические сборники литературы, периодически выпускаемые отдельными издательствами, интернет-сайты официальных организаций.

Проработка источников сопровождается выписками, конспектированием. Выписки из текста делают обычно в виде цитаты. После каждой цитаты приводится ссылка на автора и источник. Поэтому при выписке цитат и конспектировании следует делать ссылки: автор, название издания, место издания, издательство, год издания, номер страницы. Конспект может содержать в себе факты, доказательства, примеры, основные положения и выводы автора, а также личное отношение студента к изученному материалу.

Рациональной и эффективной организации исследовательской деятельности способствует составление студентом собственной картотеки проработанных по теме и смежным вопросам литературных источников. Картотека не только помогает получить представление о степени изученности данной проблемы, но и позволяет работать с литературой более целеустремленно.

Оформление курсовой работы

Оформление проекта или работы осуществляется в соответствии с требованиями положения о выполнении и защите курсовых о выполнении и защите курсовых работ (проектов).

Содержание и структура курсовых работ

Курсовая работа должна содержать элементы новизны, наряду с фундаментальным аспектом должен быть проведен анализ современного состояния изучаемой проблемы, а также отражены региональные особенности. Задание по курсовой работе необходимо индивидуализировать с учетом интересов и способностей студентов.

Курсовая работа должна состоять из введения, теоретической части, эмпирической (практической, расчетно-графической) части, заключения, списка литературы и приложения. В отдельных случаях, в соответствии с тематикой работы, эмпирическая часть может отсутствовать.

Во введении обосновывается актуальность выбранной темы исследования, задачи, цели, новизна, теоретическая и практическая значимость исследования.

Теоретическая часть должна содержать анализ состояния изучаемой проблемы на основе обзора научной, научно-информационной, справочной литературы. Представленный материал должен быть логически связан с целью исследования. В параграфах теоретической части необходимо отражать отдельные компоненты проблемы и завершать их выводами.

Эмпирическая (практическая, расчетно-графическая) часть (при наличии) включает описание системы экспериментального исследования, обоснование методов исследования, анализ результатов экспериментального исследования, схемы, графические и математические способы интерпретации полученных данных, выводы.

Заключение содержит выводы, подтверждающие или опровергающие первоначальные предположения (гипотезы), перспективы дальнейшего изучения проблемы, связь с практикой, анализ реализации целей и задач исследования.

Список литературы должен быть составлен в соответствии с требованиями к оформлению библиографий.

Приложение содержит весь фактический материал экспериментальных исследований (анкеты, опросники, схемы, чертежи, расчетные материалы, карты, рисунки, ответы респондентов и т.д.).

Содержание курсовой работы (проекта) рекомендуется представлять в объеме 20-30 страниц печатного текста. Текст работы должен быть напечатан через 1,5 интервала на одной стороне стандартного листа белой бумаги (А - 4). Текст и другие отпечатанные элементы работы должны быть черными, контуры букв и знаков – четкими, без ореола и затенения. Шрифт TimesNewRoman, кегель 14. Курсив и подчеркивание в работе не допускаются. Названия глав и параграфов выделяются полужирным шрифтом. Лист с текстом должен иметь поля: слева – 30 мм, справа – 10 мм, сверху – 20 мм, снизу – 20 мм. Нумерация страниц текста делается в правом

нижнем углу листа. Проставлять номер страницы необходимо со страницы, где печатается "Введение", на которой ставится цифра "3". После этого нумеруются все страницы, включая приложения.

Между названием главы и названием параграфа этой главы ставится пробел равный двум интервалам, а название параграфа не должно отделяться от текста этого параграфа пробелом. Названия параграфов отделяются от текста предыдущего параграфа пробелом, равным двум интервалам. Каждая глава, а также введение, выводы, предложения и список использованной литературы начинаются с новой страницы. Слово "Глава" не пишется. Главы имеют порядковые номера в пределах всей работы, обозначаемые арабскими цифрами (например: 1,2,3), после которых ставится точка. Слово "параграф" или значок параграфа в названии не ставятся. Параграфы имеют порядковые номера в пределах глав, обозначаемые арабскими цифрами (например: 1.1. и 1.2.). Заголовки глав и параграфов в тексте работы должны располагаться по центру, точку в конце названия главы и параграфа не ставят. Не допускается переносить часть слова в заголовке.

Нумерация таблиц и рисунков может быть сквозной или соотноситься с номером главы и параграфа. Например, если таблица или рисунок включены в текст первого параграфа второй главы, нумерация следующая: Таблица 2.1.1., рис. 2.1.1. Последняя цифра означает порядковый номер таблицы (или рисунка) в данном параграфе. Таблица помещается в качестве следующей страницы после первого упоминания о ней в тексте.

В рамках отведенного для руководства курсовыми работами времени регулярно проводятся индивидуальные консультации. Руководитель работы осуществляет предварительный просмотр выполненных заданий. После проверки выполнения студентом одного этапа работы, руководитель работы разрешает студенту перейти к следующему.

1.3 Заключительный этап

Работа сдается руководителю. Если работа удовлетворяет требованиям, предъявляемым к курсовым работам, она допускается к защите.

Защита курсовой работы является обязательной формой проверки выполнения работы. Защита производится на заседании кафедры, научно-методического семинара кафедры, научной проблемной группы при непосредственном участии руководителя, в присутствии студентов. Результаты наиболее интересных курсовых работ могут быть доложены на научных конференциях.

Защита состоит в коротком докладе студента по выполненной работе и в ответах на вопросы присутствующих на защите.

Результаты защиты курсовой работы, согласно действующему Положению о текущем контроле и промежуточной аттестации, оцениваются дифференцированной отметкой по пятибалльной системе.

Студент, не представивший в установленный срок работу, получивший неудовлетворительную оценку на защите или не явившийся на защиту по неуважительной причине, считается имеющим академическую задолженность.

Для ликвидации академической задолженности студент обязан в установленные сроки представить курсовую работу руководителю и защитить её перед комиссией.

Допуск к защите курсовой работы

Допуск к защите осуществляет научный руководитель курсовой работы.

Работа не допускается к защите, если она не носит самостоятельного характера, списана из литературных источников или у других авторов, если основные вопросы не раскрыты, изложены схематично, фрагментарно, в тексте содержатся ошибки, научный аппарат оформлен неправильно, текст написан небрежно.

Неудовлетворительно выполненная работа подлежит переработке в соответствии с замечаниями преподавателя, содержащимися в отзыве.

При получении допуска к защите студент вправе получить дополнительную консультацию по предстоящей защите.

Курсовая работа, студенту не возвращается и хранится на кафедре в течение 2-х лет.

После получения допуска к защите, студент самостоятельно готовится к защите: составляет текст доклада, реагирует на замечания руководителя.

Подготовка к защите курсовой работы включает устранение ошибок и недостатков, изучение дополнительных источников, готовность объяснить любые приведенные в работе положения.

В ходе защиты курсовой работы задача студента - показать углубленное понимание вопросов конкретной темы, хорошее владение материалом по теме.

Защита работы производится в соответствии с планом приёма защит курсовых работ. С ним можно ознакомиться на стенде кафедры или непосредственно уточнить у научного руководителя время и место защиты.

По результатам защиты курсовой работы в зачётную ведомость (для защиты курсовой работы) выставляется оценка.

Критерии оценки курсовой работы:

- «**отлично**» выставляется студенту, показавшему глубокие знания, которые применяются при самостоятельном исследовании избранной темы, умение обобщать практический материал и делать на основе анализа выводы. Курсовая работа с оценкой «отлично» может быть рекомендована на конкурс студенческих работ, использована в качестве доклада на научно-студенческой конференции.
- «**хорошо**» выставляется студенту, показавшему в работе и при её защите полное знание материала, всесторонне осветившему вопросы темы, но не полной мере проявившему самостоятельность в исследовании.
- **«удовлетворительно»** выставляется студенту, раскрывшему в работе основные вопросы избранной темы, но не проявившему самостоятельности в анализе или допустившему отдельные неточности содержания работы.
- «**неудовлетворительно»** выставляется студенту, не раскрывшему основные положения избранной темы и допустившему грубые ошибки в содержании работы.

2. СОДЕРЖАНИЕ КУРСОВОЙ РАБОТЫ

2.1 Структурные элементы курсовой работы

Структурными элементами курсовой работы являются:

1) титульный лист,
2) оглавление,
5) введение;
6) основная часть;
7) заключение;
8) список использованных источников;
9) приложения.

Титульный лист является первой страницей курсовой работы.

Оглавление включает введение, наименование всех глав и параграфов, заключение, список использованных источников и приложения с указанием номеров страниц, с которых начинаются эти элементы работы. Следует обратить внимание, что через оглавление прослеживается структура всей работы. Желательно основную часть работы уместить в 2 главы, каждая из которых разбивается на 2-3 параграфа. Глава не может содержать один параграф. Наличие в главе более трех параграфов говорит о поверхностном представлении данных вопросов, так как к работе предъявляются жесткие требования по объему работы.

Введение. Во введении обосновывается **актуальность темы**, **формулируются цель** курсовой работы **и комплекс взаимосвязанных задач**, подлежащих решению в процессе работы, а также представляется **теоретическая и методологическая база.**

Актуальность темы отражает степень важности её раскрытия на данный момент, то есть, почему читателю будет интересно, важно или просто необходимо познакомиться с Вашей работой. Это 2-3 предложения, которые анонсируют Вашу работу. Важно во введении**,** раскрывая актуальность темы, не уйти в раскрытие заявленной проблемы. Не следует писать фразы типа: «Мне эта тема очень интересна, поэтому она актуальна». Здесь необходимо заинтересовать читателя предметно и содержательно, а не своим примером.

Цель работы – это результат, к которому Вы желаете прийти и путь к нему. Цель работы одна. Формулируем её с использованием неопределённой формы глагола: изучить, исследовать, проанализировать, рассмотреть и т.д. Цель вашей курсовой работы скрыта в названии работы.

Задачи работы отражают шаги, делая которые, Вы достигаете цели. Вы должны поставить 6 - 7 задач. Они фактически связаны с параграфами Вашей работы. Задачи формулируются аналогично цели с использованием неопределенной формы глагола: изучить, исследовать, проанализировать, рассмотреть, оценить, выявить и т.д.

Теоретическая и методологическая база подразумевает краткое описание использованных источников с указанием основных авторов. Это «поклон» в адрес людей, чьими трудами Вы воспользовались, а также возможность для специалиста с первого взгляда понять, о чем написана работа и какого подхода к данной проблеме придерживаетесь Вы.

Объем этой части работы 1 лист.

Так как введение – это та часть работы, которая подразумевает 100% самостоятельное изложение, важно не перейти на «ты» в общении с подразумеваемым читателем, то есть **всё изложение материала должно осуществляется в неопределенной форме глагола**. Нельзя употреблять местоимение «я».

Основная часть. Требования к содержанию основной части работы даны ранее.

Заключение. В заключении формулируются краткие выводы, полученные в ходе выполнения поставленных задач и цели. Вы кратко излагаете то, что написали в работе, но это самое главное, что вы хотите донести до читателя, самый «сок» Вашего труда. В заключение можно вынести числовые данные в целях иллюстрации вывода. Они должны отражать цель и задачи исследования, и не быть «вырванными» из работы числами.

Список использованных источников должен содержать сведения об источниках, использованных при выполнении работы.

Приложения. В приложении рекомендуется включать материалы, связанные с выполненной работой, которые по каким-либо признакам не могут быть включены в основную часть.

2.2 Содержание основной части курсовой работы

Теоретический раздел

Раздел должен иметь наименование, отражающее теоретические аспекты темы курсовой работы. В общем случае в этом разделе должны быть отражены:

1) характеристика предмета исследования в соответствии темой курсовой работы, в частности, должен присутствовать анализ понятийной базы по заявленной теме.

2) **анализ состояния вопроса**, являющегося темой работы, при необходимости – методы анализа.

3) основные подходы к решению проблемы, являющейся темой курсовой работы.

Теоретический раздел должен давать ответы на вопросы, что представляет собой проблема, являющаяся темой курсовой работы.

Объем раздела – 10 - 12 с. В разделе должно быть **не менее двух и не более трех подразделов**.

Аналитический раздел

Наименование данного раздела должно быть связано с анализом состояния предмета исследования, являющегося темой курсовой работы.

Исследование проблемы должно быть целевым и глубоким.

В аналитическом разделе используются различные приемы анализа, указываются источники информации, приводятся аналитические таблицы и расчеты, графические способы отражения результатов анализа, делаются выводы.

В данном разделе обязательно использование не только статистического материала, но и результатов анализа, проведенного специалистами по данному вопросу, которые изложены в источниках периодической печати.

Объем аналитического раздела курсовой работы не должен превышать 10-12 страниц.

СПИСОК РЕКОМЕНДУЕМОЙ ЛИТЕРАТУРЫ

Перечень основной литературы:

1. Галатенко В.А. Основы информационной безопасности. – М.: Интернет-Университет Информационных Технологий, 2020.
2. Петраков А.В. Защита информации в компьютерных системах и сетях. – М.: Радио и связь, 2021.
3. Шаньгин В.Ф. Комплексная защита информации в корпоративных системах. – М.: ДМК Пресс, 2019.
4. Лопатин В.Н. Информационная безопасность России. – СПб.: Юридический центр Пресс, 2018.
5. Белов Е.Б., Лось В.П. Основы информационной безопасности. – М.: Горячая линия – Телеком, 2020.

Перечень дополнительной литературы:

6. Федеральный закон от 27.07.2006 № 149-ФЗ "Об информации, информационных технологиях и о защите информации".
7. Федеральный закон от 27.07.2006 № 152-ФЗ "О персональных данных".
8. Федеральный закон от 26.12.1995 № 208-ФЗ "О безопасности".
9. Приказы ФСТЭК России
10. ГОСТ Р 50922-2006 "Защита информации. Основные термины и определения".
11. ГОСТ Р 57580.1-2017 "Безопасность финансовых организаций".