МИНИСТЕРСТВО ТРАНСПОРТА И КОММУНИКАЦИЙ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования

«Белорусский государственный университет транспорта»

Кафедра «Информационно-управляющие системы и технологии»

Отчет по практическим работам  
по дисциплине «Основы алгоритмизации и программирования»

Выполнил Проверил

студент группы ГИ-11 ст. преп. каф. ИУСиТ

Зайцев И. С. Голдобина Т. А.

Гомель, 2019

Содержание

[Практическая работа №2 Знакомство со средой программирования. Разработка, отладка и выполнение простейшей программы 4](#_Toc28688669)

[Задания 4](#_Toc28688670)

[Контрольные вопросы 7](#_Toc28688671)

[Практическая работа №3 Разработка, отладка и выполнение программы на составление и вычисление выражений. Преобразование типов данных 9](#_Toc28688672)

[Задания 9](#_Toc28688673)

[Контрольные вопросы 12](#_Toc28688674)

[Практическая работа №4 организация ввода/вывода данных. Формированный ввод/вывод 14](#_Toc28688675)

[Задания 14](#_Toc28688676)

[Контрольные вопросы 19](#_Toc28688677)

[Практическая работа №5 Разработка алгоритма, составление, отладка и выполнение программы линейного вычислительного процесса 21](#_Toc28688678)

[Задания 21](#_Toc28688679)

[Контрольные вопросы 25](#_Toc28688680)

[Практическая работа №6 Применение операций сравнения, побитовых и логических операций 27](#_Toc28688681)

[Задания 27](#_Toc28688682)

[Контрольные вопросы 30](#_Toc28688683)

[Практическая работа №7 Разработка алгоритма, составление, отладка и выполнение программы с ветвлением 31](#_Toc28688684)

[Задания 31](#_Toc28688685)

[Контрольные вопросы 37](#_Toc28688686)

[Практическая работа №8 Разработка алгоритма, составление, отладка и выполнение циклической программы с известным числом повторений 40](#_Toc28688687)

[Задания 40](#_Toc28688688)

[Контрольные вопросы 44](#_Toc28688689)

[Практическая работа №9 Разработка алгоритма, составление, отладка и выполнение программы с использованием итерационных циклов 46](#_Toc28688690)

[Задания 46](#_Toc28688691)

[Контрольные вопросы 53](#_Toc28688692)

[Практическая работа №10 Разработка и выполнение программы с использованием разветвлений и вложенных циклов 55](#_Toc28688693)

[Задания 55](#_Toc28688694)

# Практическая работа №2 Знакомство со средой программирования. Разработка, отладка и выполнение простейшей программы

### **Цель**

Познакомиться со средой программирования. Разработать, произвести отладку и выполнить простейшие программы.

## Задания

Задание 1

### Условие

1. Ознакомиться со средой программирования Code::Blocks.
2. Запустите среду программирования Code::Blocks.
3. Создайте новый проект. Сохраните его с именем task1 в папке pr1\_1.
4. Внимательно ознакомьтесь с составом главного меню и панелей инструментов.
5. Откройте файл main.c. Выполните компиляцию и прогон программы. В дальнейшем выполняйте компиляцию и прогон после каждого изменения и, при необходимости, исправляйте ошибки.
6. C помощью функции printf() выведите свою фамилию, имя, отчество на латинице.
7. C помощью функции printf() выведите свою фамилию, имя, отчество на русском. Что получилось? Локализуйте кириллицу в консоли и выведите информацию корректно.
8. Объявите целочисленную переменную x. Присвойте ей значение 5. Выведите в консоли с помощью функции printf().
9. Инициализируйте переменную у значением 2.
10. Попробуйте вывести сумму, произведение, разность и частное этих переменных, каждое в отдельной строке. Что получилось?
11. Дополните вывод поясняющим текстом. Например, чтобы получилось "Сумма x и y равна ...(тут рассчетное значение)". Или просто "x + у = ...". Пробелы при выводе тоже учитывать.
12. Доведите версию программы до идеального состояния (логично, корректно и красиво). Вот её-то вместе с результатами и надо поместить в отчет.
13. Сохраните проект и закройте.

### Программный код

#include <stdio.h>

#include <locale.h>

int main()

{

int x = 5, y = 2, c;

setlocale (0,"Russian");

printf ("Zaytsev Igor Sergeevich\n");

printf ("Зайцев Игорь Сергеевич\n");

printf ("x = %d, y = %d\n", x, y);

c = x + y;

printf("%d\n", c);

c = x - y;

printf("%d\n", c);

c = x \* y;

printf("%d\n", c);

c = x / y ;

printf("%d\n", c);

return 0;

}

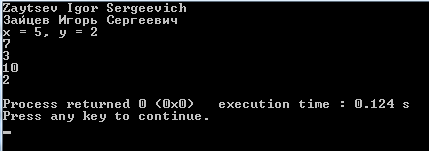


Рисунок 1.1 – Резульатат задания 1

Задание 2

### Условие

Есть фрагмент программного кода:

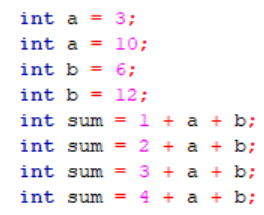


Рисунок 1.2 – Фрагмент программного кода

Необходимо:

Закомментировать некоторые строки так, чтобы программа компилировалась без ошибок.

Получить в результате значение sum = 12.

Вывести значения переменных и расчётную сумму.

### Программный код

#include <stdio.h>

#include <stdlib.h>

int main()

{

int a = 3;

//int a = 10;

int b = 6;

//int b = 14;

/\*int sum = 1 + a + b;

int sum = 2 + a + b;\*/

int sum = 3 + a + b;

//int sum = 4 + a + b;

printf ("sum = %d\n", sum);

printf ("%d\n",a);

printf ("%d\n",b);

printf ("%d\n",sum);

return 0;

}

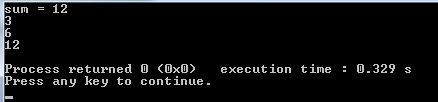


Рисунок 1.3 – Результат задания 2

Задание 3

### Условие

Есть фрагмент программного кода:

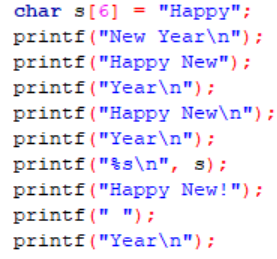


Рисунок 1.4 – Фрагмент программного кода

Необходимо:

Закомментировать некоторые строки так, чтобы программа компилировалась без ошибок.

Получить в консоли строку "Happy New Year!"

### Программный код

#include <stdio.h>

#include <stdlib.h>

int main()

{

//char s[6] = "Happy";

//printf("New Year\n");

printf("Happy New");

//printf("Year\n");

//printf("Happy New\n");

//printf("Year\n");

//printf("%s\n", s);

//printf("Happy New!");

printf(" ");

printf("Year\n");

return 0;

}

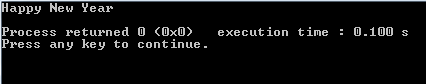


Рисунок 1.5 – Результат задания 3

## Контрольные вопросы

1. Что означает #include?

Директива, которая подключает заголовочные файлы.

1. Какой заголовочный файл содержит локальные культуры?

locale.h

1. Что означает int main(void)?

Главная функция, которая возвращает целое значение.

1. Какая функция локализует кириллицу в консоли?

setlocale()

1. Чем объявление переменной отличается от инициализации?

Объявление переменной приводит к выделению памяти, которая при этом никак не очищается, и значение переменной не определено. Инициализация приводит к тому, что при выделении ячейки памяти в неё записывается начальное значение.

1. Спецификатор вывода целого числа.

%d

1. Спецификатор вывода вещественного числа.

%f

1. Как выполнить в ходе вывода перевод на новую строку?

printf( “…\n”).

### **Выводы по работе**

В этой практической работе мы познакомились со средой программирования. Разработали, произвели отладку и выполнили простейшие программы.

# Практическая работа №3 Разработка, отладка и выполнение программы на составление и вычисление выражений. Преобразование типов данных

### **Цель**

Познакомиться со средой программирования. Разработать, произвести отладку и выполнить программы на составление и вычисление выражений. Научиться преобразовывать типы данных.

## Задания

### **Задание 1**

### Условие

Составить линейную программу, в которой вычисляется значение выражения:  
C:\Temp\Rar$EXa0.441\oap_gi\lab3\files\image020.png

1. Проект сохранить с именем task3\_1 в папке pr3\_1.
2. Целочисленные значения переменных, входящих в выражение, задать непосредственно в коде программы
3. Организовать вычисление выражения в операторе присваивания
4. Выполнить программу и получить результат, исправив ошибки при их наличии

### Программный код

#include <stdio.h>

#include <stdlib.h>

#include <locale.h>

#include <math.h>

int main()

{

int x, y, z, v;

setlocale (LC\_ALL,"Russian");

printf("Введите x, y, z\n");

scanf("%d%d%d", &x, &y, &z);

v = ((x\*y)\*pow(z,3))+ ((2- pow((x-y),3))) / abs(5 - pow(x,2)\*y) + pow(x+y+y,x\*y-z);

printf("Ответ %d", v);

return 0;

}

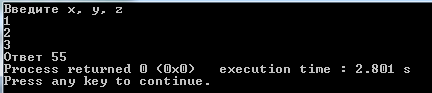


Рисунок 1.1 – Результат задания 1

### **Задание 2**

### Условие

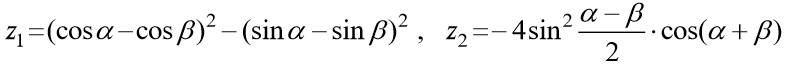
Составить программу для расчета выражений z1 и z2, результаты которых должны совпадать.  


Рисунок 1.2 – Фрагмент программного кода

### Программный код

#include <stdio.h>

#include <stdlib.h>

#include <math.h>

#define PI 3.1415926

int main()

{int b, a;

float z1, z2;

printf("Enter\n");

scanf("%d%d", &b, &a);

z1= pow(cos(a) - cos(b),2) - pow(sin(a) - sin(b),2);

z2= (-4.) \* pow(sin((a-b)\*1./2.),2) \* cos(a+b);

printf("answers: %f; %f\n", z1, z2);

return 0;

}

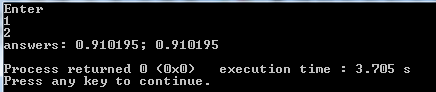


Рисунок 1.3 – Результат задания 2

Задание 3\*

### Условие

Составить программу, которая выводит на экран размер типов данных int, long int, long long int, short, char, float, double, long double. При определении размера типов использовать функцию sizeof().

### Программный код

#include <stdio.h>

#include <stdlib.h>

int main()

{

printf("int = %d\n", sizeof(int));

printf("long int = %d\n", sizeof(long int));

printf("long long int = %d\n", sizeof(long long int));

printf("short = %d\n", sizeof(short));

printf("char = %d\n", sizeof(char));

printf("float = %d\n", sizeof(float));

printf("double = %d\n", sizeof(double));

printf("long double = %d\n", sizeof(long double));

return 0;

}



Рисунок 1.4 – Результат задания 3

Задание 4\*

### Условие

Вы пришли в обменный пункт валюты с суммой в n долларов и хотите поменять эту сумму на евро, но банк не проводит прямую конвертацию долларов в евро. Сначала доллары переводятся в рубли, а затем рубли – в евро. Составить программу на С, которая считывает с клавиатуры количество долларов и переводит в рубли, выводит это количество на экране, а затем вычисляет целое количество евро, которые мы можем получить при обмене и остаток рублей. Курс для перевода доллара в рубли и курс для перевода евро в рубли задается с помощью директивы препроцессора #define.

### Программный код

#include <stdio.h>

#include <stdlib.h>

#include <math.h>

#include <locale.h>

#define d 2.075

#define c 2.26

int main()

{

setlocale(LC\_ALL,"");

int y;

float n, x, o;

printf("Введите количество долларов:\n");

scanf("%f",&n);

x=n\*d;

printf("%f рублей\n", x);

y=x/c;

o=(x/c-y)\*c;

printf("У вас будет %d евро и %f рублей\n", y, o);

return 0;

}

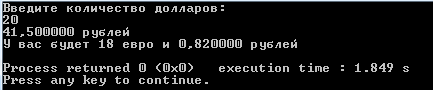


Рисунок 1.5 – Результат задания 4

## Контрольные вопросы

1. Что означает #define?

#define – это директива препроцессора, которая создает макросы или пре-процессорные идентификаторы.

2. Какой размер в вашей ОС имеет тип данных short ?

В мой операционной системе тип данных short имеет размер 2 байта.

3. Как задать константу типа long double ?

Константу типа long double можно задать с помощью классификатора const.1

4. Чем отличается x++ от ++x?

В первом случае инкремент имеет постфиксную форму, т.е. инкремент вы-полняется в самом конце, после того как с ним произведут операции другие опе-раторы в этом выражении, во втором случает имеет префиксную форму, в этом случае инкремент имеет больший приоритет чем другие операции, и он выпол-нится первым.

5. Что такое декрементация?

Декрементация – это унарная операция, которая заключается в вычитании единицы из числа.

6. Приоритет выполнения арифметических операций.

Наивысший приоритет арифметических операций имеют: инкремент, декре-мент, унарный минус, умножение, деление и остаток от деления. Низший прио-ритет имеют операции сложения и вычитания.

Операции с одинаковым приоритетом выполняются слева направо. Исполь-зую круглые скобки можно изменить порядок вычисления.

7. Что означает операция %?

Результатом такой операции является остаток от деления целых чисел.

### **Выводы по работе**

Познакомился со средой программирования. Разработал, произвел отладку и написал программы на составление и вычисление выражений. Научился преобразовывать типы данных.

# Практическая работа №4 организация ввода/вывода данных. Формированный ввод/вывод

### **Цель**

Познакомиться с организацией ввода/вывода данных и составить программы, их обеспечивающие. Изучить форматированный ввод/вывод.

## Задания

### **Задание 1**

### Условие

### Составить программу, которая выводит форматированный текст, представленный ниже (пробелы учитываются!).

15/60\*100%=

25%

"H"\

Проект сохранить с именем task4\_1 в папке pr4\_1.

Выполнить программу и получить результат, исправив ошибки при их наличии

### Программный код

[#include](https://vk.com/im?sel=247681002&st=%23include) <stdio.h>  
[#include](https://vk.com/im?sel=247681002&st=%23include) <locale.h>  
[#include](https://vk.com/im?sel=247681002&st=%23include) <math.h>  
  
int main()  
{  
float x = 15, y = 60;  
int c = 25;  
setlocale(LC\_ALL,"");  
printf("15/60\*100%%=\n");  
printf("%13d%%\n",c );  
printf("\"H\"\\");  
  
return 0;  
}

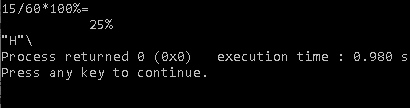


Рисунок – Результат задания 1

### **Задание 2**

### Условие

Составить программу, в которой выводятся символы, коды которых равны

141, 103 в восьмеричной системе

65, 61 в десятичной системе

47, 43 в шестнадцатеричной системе

Добавить фрагмент, в котором вводится символ (буква) и выводится код символа нажатой клавиши.

### Программный код

[#include](https://vk.com/im?sel=247681002&st=%23include) <stdio.h>  
[#include](https://vk.com/im?sel=247681002&st=%23include) <stdlib.h>  
[#include](https://vk.com/im?sel=247681002&st=%23include) <math.h>  
  
int main ()  
{int c;  
  
  
printf("%c\n", 0141);  
printf("%c\n", 0103);  
printf("%c\n", 65);  
printf("%c\n", 61);  
printf("%c\n", 0x47);  
printf("%c\n", 0x33);  
  
  
  
scanf("%c", &c);  
printf("%d", c);  
return 0;  
}

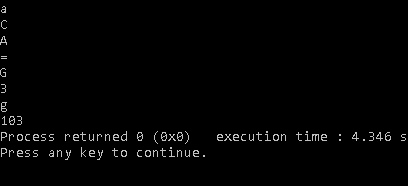


Рисунок - Результат задания 2

### Задание 3

### Условие

Составить программу, которая позволяет вводить время в формате hh:mm:ss , а выводить в формате hh ч mm мин ss с .  
Ввод выполнить в одном операторе scanf(), вывод - в одном операторе printf().

Проект сохранить с именем task4\_3 в папке pr4\_3.

### Программный код

[#include](https://vk.com/im?sel=247681002&st=%23include) <stdio.h>  
[#include](https://vk.com/im?sel=247681002&st=%23include) <stdlib.h>  
[#include](https://vk.com/im?sel=247681002&st=%23include) <locale.h>  
  
int main()  
{int h, m, s;  
setlocale(LC\_ALL,"");  
printf ("Введите дату:\n");  
scanf ("%2d%\*c%2d%\*c%2d",&h, &m, &s);  
printf ("%d ч %d мин %d сек",h, m, s);  
  
return 0;  
}

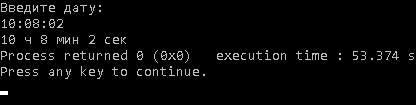


Рисунок - Результат задания 3

### **Задание 4**

### Условие

Написать программу на языке С, которая считывает с клавиатуры переменные следующих типов:

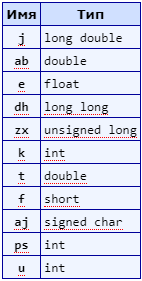


Рисунок - Фрагмент программного кода

Вывести на экран переменные с заданной шириной поля и точностью в требуемой системе счисления



Рисунок - Фрагмент программного кода

Вывести на экран переменные с переменными шириной поля и точностью заданных в соответствующих переменных

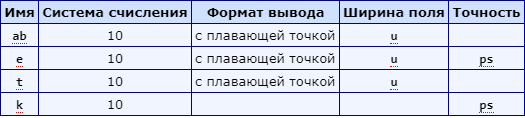


Рисунок - Фрагмент программного кода

Программный код

#include <stdio.h>

#include <stdlib.h>

#include <math.h>

int main()

{

long double j;

double ab;

float e;

long long dh;

unsigned long zx;

int k;

double t;

short f;

signed char aj;

int ps;

int u;

scanf("%Lf\n", &j);

scanf("%lf\n", &ab);

scanf("%f\n", &e);

scanf("%lli\n", &dh);

scanf("%lu\n", &zx);

scanf("%d\n", &k);

scanf("%lf\n", &t);

scanf("%d\n", &aj);

scanf("%hi\n", &f);

scanf("%d\n", &ps);

scanf("%d", &u);

printf("%.2Lf\n", j);

printf ("%8.3g\n", ab);

printf ("%8.3f\n", ab);

printf ("%8.3e\n", ab);

printf ("%f\n", e);

printf ("%6f\n", t);

printf ("%x\n", zx);

printf ("%6lu\n", zx);

printf ("%10o\n", zx);

printf ("%lli\n", dh);

printf ("%.5d\n", k);

printf ("%hi\n", f);

printf ("%d\n", aj);

printf("%\*f\n", u, ab);

printf("%\*.\*f\n", u, ps, e);

printf("%\*f\n", u, t);

printf("%.\*d\n", ps, k);

return 0;}



Рисунок - Результат задания 4

## Контрольные вопросы

1. Для чего предназначена функция puts()?— Puts() – вывод строки с переходом на начало новой строки.

2. Как вывести значение в в виде десятичного числа без знака ?

%d

3. Как вывести символ %? — %%.

4. Как выполнить модификацию формата вывода с заданной точностью для вещественных чисел ? %.xd (x – заданная точность)

5. Что необходимо сделать в программе для вычисления выражений с математическими функциями? — Требуется подключение заголовочного файла math.h.

6. Перечислите и определите обратные тригонометрические функции.

Arcsin, arccos, arctg, arcctg

7. Функция scanf(). Общий формат использования. — Функция scanf()является процедурой ввода общего назначения, считывающей данные из потока stdin. Она может считывать данные всех базовых типов и автоматически конвертировать их в нужный внутренний формат. Scanf ("...спецификатор формата... ", &адресс вводимой переменной).

8. Какие элементы сможет включать управляющая строка scanf()? — Управляющая строка состоит из символов, разделённых на 3 категории: спецификаторы формата, разделители и символы, не являющиеся разделителями (за исключением ‘%’).

9. Разделители ввода и подавление присваивания при использовании функции scanf(). — Разделители: пробел, символ прехода на новую строку, символ табуляции, символ вертикальной табуляции и символ прогона бумаги. Символ подавления присваивания – \*.

10. Функция gets(). Очистка потока. — Функция gets() считывает строку S символов, пока не будет нажата клавиша Enter.

### **Выводы по работе**

Познакомились с основными возможностями организации ввода/вывода. Форматированным вводом и выводом данных, потоками ввода/вывода, программированием линейных алгоритмов.

# Практическая работа №5 Разработка алгоритма, составление, отладка и выполнение программы линейного вычислительного процесса

### **Цель**

Разработать алгоритм и выполнить программы линейного вычислительного процесса.

## Задания

### **Задание 1**

### Условие

Составить блок-схему алгоритма и линейную программу, которая позволяет ввести значения переменных и вычислить указанные выражения по формулам.

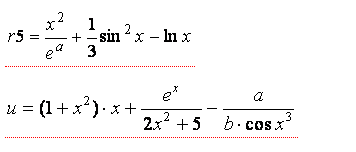


Рисунок – Фрагмент программного кода

### 

Рисунок – Блок-схема

### Программный код

#include <stdio.h>

#include <stdlib.h>

#include <math.h>

#define e 2,71828

int main()

{int x, a, b;

float r5, u;

printf("Enter\n");

printf("x=");

scanf("%d",&x);

printf("a=");

scanf("%d",&a);

printf("b=");

scanf("%d",&b);

r5=(pow(x,2)/exp(a))+ 1/3.\*pow(sin(x),2)-log(x);

u=((1+(pow(x,2)))\*x)+exp(x)/(2\*pow(x,2)+5)- a/b\*cos(pow(x,3));

printf("r5=");

printf("%f\n",r5);

printf("u=");

printf("%f",u);

return 0;

}

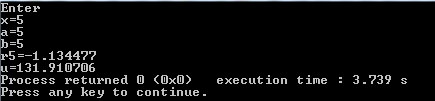


Рисунок – Результат задания 1

### **Задание 2**

### Условие

Написать программу, которая запрашивает 8 целых чисел в следующем формате:

43 7 17 34 8 34 3 12

Для считывания используется единственная функция scanf(). При этом считываются 1-е, 3-е, 5-е и 6-е значения и соответственно записываются в переменные**oh, co, m, u.**  
Определить с помощью возвращаемого значения функции scanf() и вывести на экран количество успешно считанных значений.

Вывести на экран с помощью функции printf() значение переменной**oh**три раза: до применения, с применением операции ***префиксной формы инкремента*** и после нее.  
Вывести на экран с помощью функции printf() значение переменной**co**три раза: до применения, с применением операции ***постфиксной формы инкремента*** и после нее.  
Вывести на экран с помощью функции printf() значение переменной**m**три раза: до применения, с применением операции ***префиксной формы декремента*** и после нее.  
Вывести на экран с помощью функции printf() значение переменной**u**три раза: до применения, с применением операции ***постфиксной формы декремента*** и после нее.

### Программный код

#include <stdio.h>

#include <stdlib.h>

int main()

{int oh, co, m, u, x;

x=scanf("%d %\*d %d %\*d %d %d %\*d %\*d",&oh, &co, &m, &u);

printf ("%d %d %d %d\n",oh,co,m,u);

printf("x=%d\n",x);

printf ("oh=%d\n",oh);

printf ("oh=%d\n",++oh);

printf ("oh=%d\n\n",oh);

printf ("co=%d\n",co);

printf ("co=%d\n",co++);

printf ("co=%d\n\n",co);

printf ("m=%d\n",m);

printf ("m=%d\n",--m);

printf ("m=%d\n\n",m);

printf ("u=%d\n",u);

printf ("u=%d\n",u--);

printf ("u=%d\n\n",u);

return 0;}



Рисунок - Результат задания 2

### Задание 3

### Условие

Написать программу, которая запрашивает 7 целых чисел в следующем формате:

34!7@ 27;3 98:34 #9

Для считывания используется единственная функция scanf(). При этом считываются 1-е, 2-е, 4-е и 7-е значения и соответственно записываются в переменные**co, me, h, ere.**  
Определить с помощью возвращаемого значения функции scanf() и вывести на экран количество успешно считанных значений.  
Вывести значения всех считанных переменных на экран.  
С помощью сокращенной операции вычитания из переменной **me**вычесть значение переменной**h.**  
Поменять местами значения переменных**co**и**ere.**  
Вывести новые значения переменных на экран.

### Программный код

#include <stdio.h>

#include <stdlib.h>

int main()

{int co, me, h, ere, x, p;

x=scanf("%2d%\*[!]%1d%\*[@ 27 ;]%1d %\*[98]%\*[:, , 34, #]%1d",&co, &me, &h, &ere);

printf("%d %d %d %d\n", co,me,h,ere);

printf("x=%d\n",x);

printf("me-h=%d\n", me-h);

p=co;

co=ere;

ere=p;

printf("%d %d", co, ere);

return 0;}

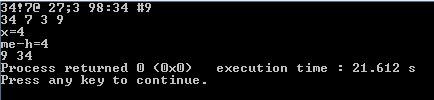


Рисунок - Результат задания 3

## Контрольные вопросы

1. Способы описания алгоритмов.  
Схематические. Псевдокод. Текстуальные.  
2. Графический способ описания алгоритмов.  
Схема программы отображает последовательность операций в программе и включает символы данных, символы процесса, линейные символы и специальные символы.  
3. Основные символы данных.

Параллелограмм - обозначает вводимые данные. Прямоугольник с волнистой нижней стороной - используется для обозначения выводимых данных.  
4. Блок, используемый для вывода информации в документ.  
Символ данных «Документ», представленный в виде прямоугольника с волнистой нижней стороной.  
5. Символ процесса.  
Прямоугольник, который используется для обозначения обработки данных любого типа.  
6. Как перенести часть блок-схемы на новую страницу?  
С помощью специального символа «Соединитель», представленного в виде окружности.  
7. Спецификатор %p.  
Используется для ввода или вывода указателя.  
8. Спецификатор %n вывода данных.  
Используется для определения числа символов, выведенных к моменту обработки самого спецификатора.   
9. Спецификатор %n ввода данных.  
Используется для определения количества прочитанных символов.  
10. Использование набора сканируемых символов.  
%[] используется для определения набора символов, которые могут быть прочитаны функцией scanf() и присвоены соответствующему массиву символов.

### **Выводы по работе**

Составил блок-схему. Написал линейные программы которые позволяют ввести значения переменных и вычислить указанные выражения по формулам.

# Практическая работа №6 Применение операций сравнения, побитовых и логических операций

### **Цель**

### Применить операции сравнения, побитовых и логических операций.

## Задания

### **Задание**

### Условие

Объявить целочисленную переменную x. Вывести (в шестнадцатеричной системе счисления) адрес ячейки, в которой будет размещено значение этой переменной.

### Программный код

#include <stdio.h>

#include <stdlib.h>

int main()

{int x;

printf("x=%x\n",&x);

return 0;

}

  
Рисунок 1 – Решение задания 0

### Задание 1

### Условие

Составить программу на С, которая считывает 2 шестнадцатеричных числа с клавиатуры (a, b). Напечатать в десятичном и шестнадцатеричном виде

инверсию a, b

a, b, a<<10, b>>7

результат побитового умножения a<<3 и b>>7

результат побитового сложения a и b

### Программный код

#include <stdio.h>

#include <stdlib.h>

int main()

{int a, b;

printf("Enter a and b:\n");

scanf("%x%x", &a, &b);

printf("a=%d b=%d\n",~a,~b);

printf("a=%x b=%x\n\n",~a,~b);

printf("a=%d b=%d a=%d b=%d\n",a, b, a<<10, b>>7);

printf("a=%x b=%x a=%x b=%x\n\n",a, b, a<<10, b>>7);

printf("a=%d\n",a<<3 & b>>7);

printf("b=%x\n\n",a<<3 & b>>7);

printf("a=%d\n",a | b);

printf("b=%x\n\n",a | b);

return 0;

}

### Задание 2

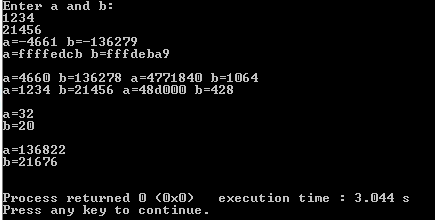


Рисунок 2 – Решение задания 1

### Условие

Ввести с клавиатуры шестнадцатеричное число и записать его в переменную a типа unsigned int.

Включить 4, 8, 10, 11, 16, 19, 23, 30 биты, не изменяя остальных, записав результат в переменную b.

Выключить 0, 4, 5, 8, 9, 13, 23, 28 биты, не изменяя остальных, записав результат в переменную c.

Переключить 0, 2, 5, 7, 8, 11, 13, 29 биты, не изменяя остальных, записав результат в переменную d.

Каждая маска должна быть представлена в виде восьмеричного, десятичного и шестнадцатеричного чисел.  
Результаты выполнения всех пунктов вывести на экран в шестнадцатеричной системе.

### Программный код

#include <stdio.h>

#include <stdlib.h>

#include <math.h>

#include <locale.h>

int main()

{unsigned int a,b,c,d;

setlocale(LC\_ALL,"");

printf("Введите a:");

scanf("%x", &a);

printf("%x\n", b=a|0x40890D10);

printf("Маска в 8с/с=%o,\nМаска в 10с/c=%d,\nМаска в 16с/с=%x\n",010042206420, 1082723600, 0x40890D10);

printf("%x\n",c=a&0x2F7FDCCE);

printf("Маска в 8с/с=%o,\nМаска в 10с/c=%d,\nМаска в 16с/с=%x\n",05737756316, 796908750, 0x2F7FDCCE);

printf("%x\n",d=a^0x200029A5);

printf("Маска в 8с/с=%o,\nМаска в 10с/c=%d,\nМаска в 16с/с=%x\n",04000024645, 536881573, 0x200029A5);

return 0;

}

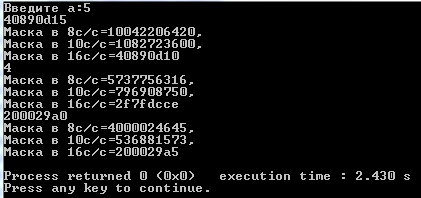
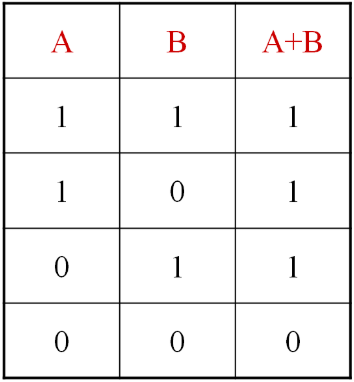


Рисунок 3 – Решение задания 2

## Контрольные вопросы

1. Особенности логических значений языков С и С++ — существуют 2 основных логических значения в данных языках. Это true и false. Которым cоответствуют значения 1 и 0, соответственно. В языке С++ для логических значений существует специальный тип — bool.
2. Как обозначается операция сравнения "равно"? — ==
3. Перечислить логические операции С — ! x — логическое отрицание (НЕ), x && y — логическое умножение (И), x || y — логические сложение (ИЛИ).
4. С операндами каких типов применимы логические операции? — int, unsigned int, long, unsigned long. В общем, целочисленные)
5. Чем побитовые операции отличаются от логических? —логические операции используются только для построения логических условий. Тогда как побитовые операции применяются в бинарной арифметике.
6. Таблица истинности побитового "или" — 
7. Что такое шаблонная маска? — битовый шаблон, используемый для получения значения бита при побитовых операциях, составленный из 0 и 1.
8. Каким символом в маске выполняется включение битов? — |
9. Приоритет логических операций и операций сравнения. — 1.!; 2.>, <, >=, =<; 3.==, !=; 4. &; 5.^; 6.|; 7.&&; 8.||

### Выводы по работе

### Применил операции сравнения, побитовых и логических операций.

# Практическая работа №7 Разработка алгоритма, составление, отладка и выполнение программы с ветвлением

### **Цель**

Разработать алгоритм и составить, отладить и выполнить программы с ветвлением.

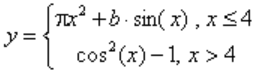
## Задания

### **Задание 1**

### Условие

### Изобразить блок-схему алгоритма и составить программу, вычисляющую значение выражения:

### Программный код

  
Рисунок 1 – Условие задания 1

#include <stdio.h>

#include <stdlib.h>

#include <math.h>

#define PI 3.1415926

int main()

{int x, b;

double y;

scanf("%d%d",&x,&b);

if (x<=4){

y=PI\*pow(x,2)+ b \* sin(x);}

else {

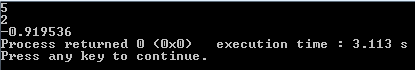
y=pow(cos(x),2)-1;}

printf("%f",y);

return 0;

}

### **Блок-схема:**

  
Рисунок 2 – Решение задания 1

### 

### Задание 2

### Условие

Составить блок-схему алгоритма и программу, в которой вычисляется значение выражения:

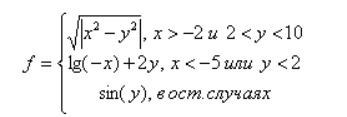


Рисунок 3 - Условие задания 2

**Программный код**

#include <stdio.h>

#include <stdlib.h>

int main()

{int x, y;

double f;

scanf("%d%d",&x, &y);

if (x>-2 && 2<y && y<10){

f= sqrt(abs(pow(x,2)-pow(y,2)));}

else if (x<-5 || y<2){

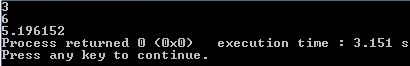
f=log10(-x)+ 2 \* y;}

else {

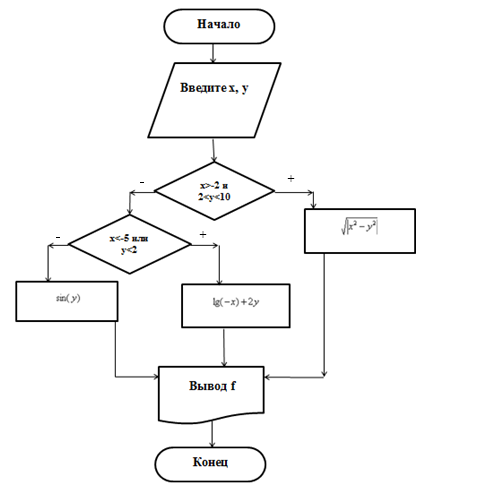
f=sin(y);}

printf("%f", f);

return 0;}

  
Рисунок 3 – Решение задания 2

### **Блок-схема:**



### Задание 3

**Условие**

Составить блок-схему алгоритма и программу для решения задачи:

Ввести номер четверти декартовой системы координат. Напечатать неравенства, задающие область значений x и y. Например, для 1-й четверти, "x > 0, y > 0". Предусмотреть возможность некорректного ввода исходных данных

**Программный код**

#include <stdio.h>

#include <stdlib.h>

int main()

{int x;

scanf("%d",&x);

switch (x){

case 1:

printf("x>0, y>0");

break;

case 2:

printf("x<0, y>0");

break;

case 3:

printf("x<0, y<0");

break;

case 4:

printf("x>0, y<0");

break;

default:

printf("NOT FOUND");

break;}

return 0;

}

  
Рисунок 4 – Решение задания 3

Блок-схема:

### 

### Задание 4

**Условие**

### Составить программу, в которой для решения задачи из задания 1 используется тернарный оператор.

**Программный код**

#include <stdio.h>

#include <stdlib.h>

#include <math.h>

#define PI 3.1415926

int main()

{int x, b;

double y;

scanf("%d%d",&x,&b);

y=(x<=4) ? PI\*pow(x,2)+ b \* sin(x) : pow(cos(x),2)-1;

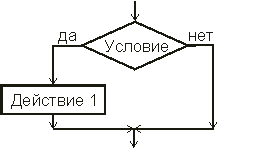
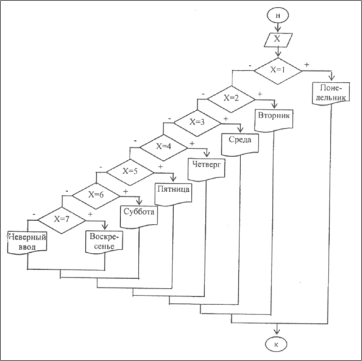
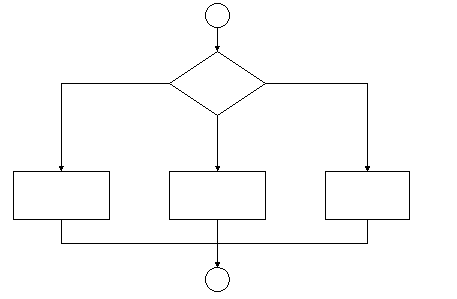
printf("%f",y);

return 0;}



Рисунок 5 - Решение задания 4

## Контрольные вопросы

1. Разветвляющиеся алгоритмы (ветвление, альтернатива) - Это алгоритмы, в которых естественное линейное следование не является обязательным
2. Блок-схема разветвляющейся вычислительной конструкции – В задании №3
3. Неполное ветвление. Понятие и блок-схема - Ветвление при выполнении определённого условия пропускается некоторый блок операций. 
4. Вложенное ветвление. Понятие и блок-схема - Любая ветвь может не быть линейным участком программы, а сама содержать ветвление. 
5. Множественное ветвление. Понятие и блок-схема - ветвление, с более чем двумя вариантами действий. 
6. Оператор ветвления в *C*. Общая форма и принцип действия - if(a==b) {

}

else {

}

Если условие в скобках после if верно, выполняется код в первых фигурных скобках. А если не верно, то во вторых скобках.

1. Неполное ветвление в C. Формат и принцип действия.

if(a==b) {

}

Если условие в скобках после if верно, выполняется код в фигурных скобках.

1. Вложенное ветвление в C. Формат и принцип действия

if(x==y) {

if(c==y) {}

else {}}

else {}

1. Множественное ветвление в C. Формат и принцип действия - Логика выполняющейся программы может быть сложнее, чем выбор одной из двух ветвей. Например, в зависимости от значения той или иной переменной, может выполняться одна из трех (или более) ветвей программы –

Switch(z){

Case 1:

//

Break;

Case 2:

//

Break;

Default:

//

Break;}

### **Выводы по работе**

Разработал алгоритм, составил, отладил и выполнил программы с ветвлением.

# Практическая работа №8 Разработка алгоритма, составление, отладка и выполнение циклической программы с известным числом повторений

### **Цель**

Разработать алгоритм, составить, отладить и выполнить циклические программы с известным числом повторений.

## Задания

### **Задание 1**

### Условие

Изобразить блок-схему алгоритма и составить программу, вычисляющую значения выражения:

при каждом значении переменной  , изменяющейся с шагом 

Использовать цикл **for**. Значения констант k1 и k2 - произвольные.

### Программный код

  
Рисунок 1 – Условие задания 1

#include <stdio.h>

#include <stdlib.h>

#include <math.h>

#define e 2,71828

int main()

{

int k1, k2;

float z,x ;

scanf("%d %d %d", &k1, &k2, &x);

printf("x\t\tz(x)\n");

for(x=1.5; x<=6; x+=0.3){

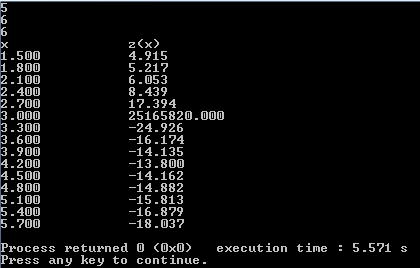
z=k1\*1./exp(x) - pow(log2(x\*1.),3) - k2\*1./(x-3);

printf("%.3f\t\t%.3f\n",x, z);

}

return 0;

}

  
Рисунок 2 – Решение задания 1

Блок-схема:

### Описание: F:\1.png

### Задание 2

### Условие

Изобразить блок-схему и составить программу, вычисляющую сумму ряда с помощью цикла **for**:

  
Рисунок 3 – Условие задания 2

### Программный код

#include <stdio.h>

#include <stdlib.h>

#include <math.h>

int main()

{scanf("%d",&x);

scanf ("%d",&m);

for (n=2 ; n<m ; n++){

y=sin(2\*n\*x)/pow(n,2)-1;

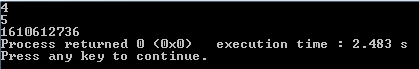
}

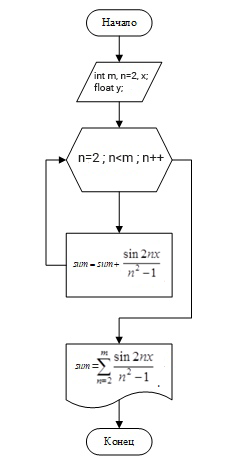
printf("%d",y);

return 0;

}

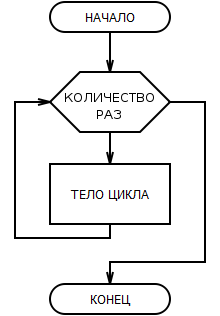
Блок-схема:

  
Рисунок 3 – Решение задания 2



## 

## Контрольные вопросы

1. Понятие цикла - любая многократно исполняемая последовательность инструкций.
2. Виды циклов – Безусловный цикл, цикл с постусловием, цикл с предусловием, цикл с известным числом повторений.
3. Циклические вычислительные конструкции – for, do while, while.
4. Цикл с известным числом повторений. Блок-схема и принцип действия - какой-либо участок программы выполняется определенное количество раз. 
5. Цикл с известным числом повторений на *С* – реализуется при помощи конструкции for.
6. Пропуск итераций – пропуск итераций можно реализовать при помощи команды continue.
7. Досрочный выход из цикла – можно реализовать при помощи команды break.

### **Выводы по работе**

Разработал алгоритм, составил, отладил и выполнил циклические программы с известным числом повторений.

# Практическая работа №9 Разработка алгоритма, составление, отладка и выполнение программы с использованием итерационных циклов

### Цель

Разработать алгоритм и составить, отладить и выполнить программы с использованием итерационных циклов.

## Задания

### Задание 1

### Условие

Изобразить блок-схему алгоритма и составить программу, вычисляющую в цикле с предусловием значение выражения:

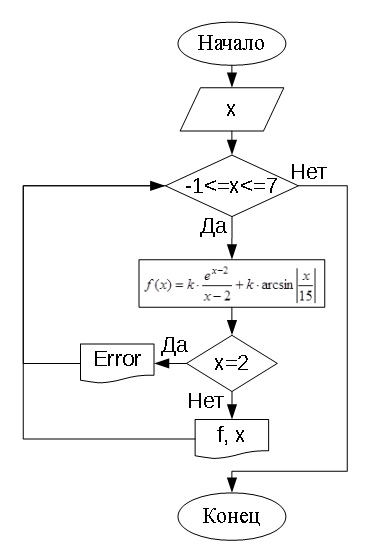
при каждом значении переменной x=[-1; 7], изменяющейся с шагом 0,35.

  
Рисунок – Условие задания 1

Значения констант - произвольные. Учесть возможность возникновения исключений во время выполнения программы.

### Блок-схема алгоритма

### Программный код

  
Рисунок – Блок-схема к заданию 1

#include <stdio.h>

#include <stdlib.h>

#include <math.h>

#define k 4

int main(){

float f, x;

printf("Enter a variable x:");

scanf ("%f", &x);

while (x>=-1 && x<=7){

f=k\*exp(x-2)/(x-2)+k\*asin(fabs(x/15));

if(x==2){

printf("Error\n");}

else{printf("f=%.3f x=%.3f\n",f,x);

}

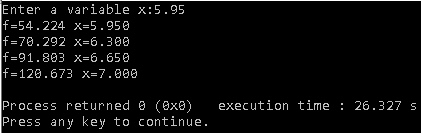
x+=0.35;

}

return 0;

**}**

### Задание 2

  
Рисунок – Решение задания 1

### Условие

Изобразить блок-схему алгоритма и составить программу, вычисляющую в цикле с постусловием значение выражения:

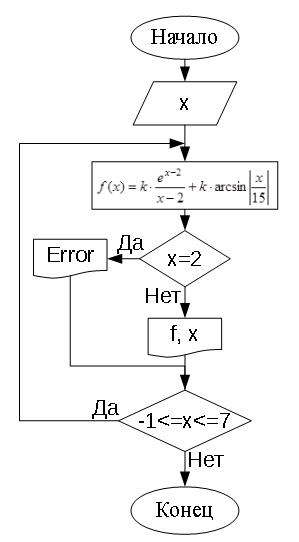
при каждом значении переменной x=[-1; 7], изменяющейся с шагом 0,35.

  
Рисунок – Условие задания 2

Значения констант - произвольные. Учесть возможность возникновения исключений во время выполнения программы.

### Блок-схема алгоритма

### Программный код

  
Рисунок – блок-схема к заданию 2

#include <stdio.h>

#include <stdlib.h>

#include <math.h>

#define k 4

int main(){

float f, x;

printf("Enter a variable x:");

scanf ("%f", &x);

do {

f=k\*exp(x-2)/(x-2)+k\*asin(fabs(x/15));

if(x==2){

printf("Error\n");}

else{printf("f=%.3f x=%.3f\n",f,x);

}x+=0.35;}

while(x>=-1 && x<=7);

return 0;}

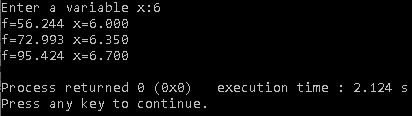


Рисунок - Решение к заданию 2

### Задание 3

### Условие

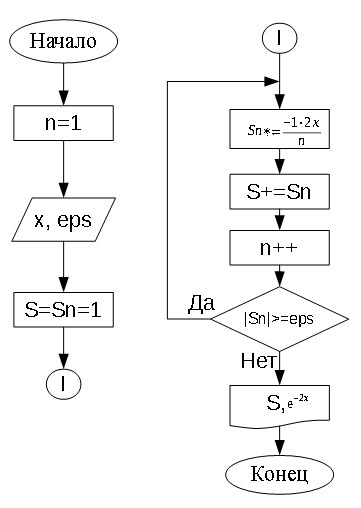
Изобразить блок-схему и составить программу, вычисляющую с помощью итерационного цикла сумму ряда:

  
Рисунок – Условие задания 3

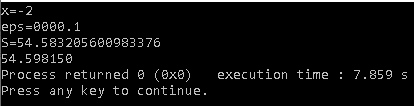
представляющего собой разложение функции в указанном интервале x=[-2; -1). Точность вычисления (eps) вводить с клавиатуры. Вывод результата в процессе отладки программы включает итоговую сумму и проверочное значение функции.

### Блок-схема алгоритма

### Программный код

  
Рисунок – Блок-схема к заданию 3

[#include](https://vk.com/im?sel=144603103&st=%23include) <stdio.h>  
[#include](https://vk.com/im?sel=144603103&st=%23include) <stdlib.h>  
[#include](https://vk.com/im?sel=144603103&st=%23include) <math.h>  
  
int main()  
{int n=1;  
double x, eps, S, Sn;  
printf("x=");  
scanf("%lf", &x);  
printf("eps=");  
scanf("%lf",&eps);  
  
  
S=Sn=1;  
do {  
Sn\*=(-1)\*2\*x/n;  
S+=Sn;  
n++;  
printf("%lf %lf\n", Sn, S);  
}  
while (fabs(Sn)>=eps);  
printf("S=%.15lf\n", S);  
printf("%lf",exp(-2\*x));  
  
return 0;}

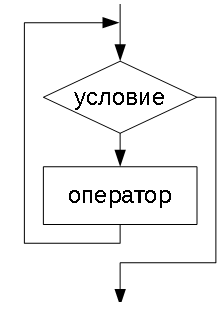
  
Рисунок – Решение задания 3

## Контрольные вопросы

1. Понятие итерационных циклов.

Итерационный цикл — это цикл, для которого число повторений тела цикла заранее неизвестно. В итерационных циклах на каждом шаге вычислений происходит последовательное приближение и проверка условия достижения искомого результата.

1. Цикл с предусловием. Блок-схема и принцип действия

Цикл с предусловием **-** цикл, который выполняется пока истинно некоторое условие, указанное перед его началом. Это условие проверяется до выполнения тела цикла, поэтому тело может быть не выполнено ни разу (если условие с самого начала ложно). 

1. Отличительные особенности цикла с предусловием

Здесь оператор (тело цикла) может быть пустым оператором, единственным оператором или блоком. Условие управляющее выражение может быть любым допустимым в языке выражением. Условие считается истинным, если значение выражения не равно нулю, а оператор выполняется, если условие принимает значение истина. Если условие принимает значение ложь, программа выходит из цикла и выполняется следующий за циклом оператор.Если условие ложно, тело цикла не будет выполнено. Благодаря этому нет необходимости вводить в программу отдельное условие перед циклом.

1. Организация цикла с предусловием на С

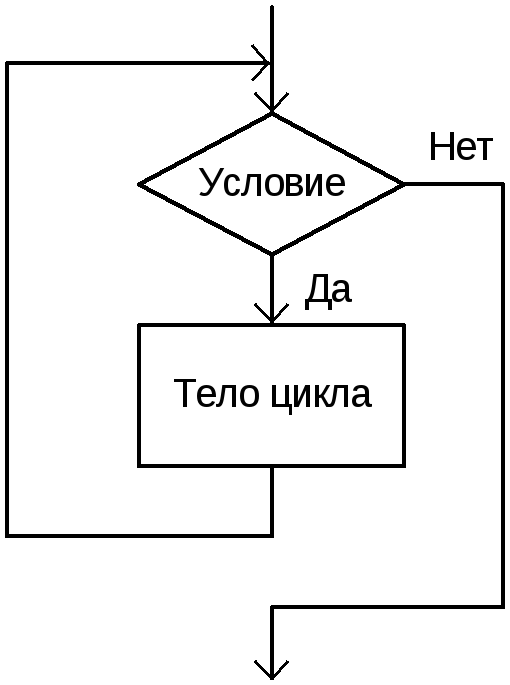
Цикл c предусловием - while.

while (условие) {оператор;}

1. Прерывание циклов

Осуществляется с помощью команды break.

1. Цикл с постусловием. Блок-схема и принцип действия

Цикл с постусловием — цикл, в котором условие проверяется после выполнения тела цикла. Отсюда следует, что тело всегда выполняется хотя бы один раз. 

1. Отличительные особенности цикла с постусловием

Главной особенностью цикла с постусловием является, то, что цикла выполнится хотя бы один.

1. Организация цикла с постусловием на С

Цикл с постусловием - do-while.

do {

оператор;

} while (условие);

### **Выводы по работе**

Разработал алгоритм и составил, отладил и выполнил программы с использованием итерационных циклов. Составил блок-схемы к ним.

# Практическая работа №10 Разработка и выполнение программы с использованием разветвлений и вложенных циклов

### Цель

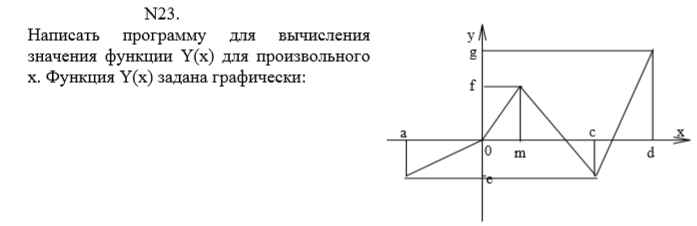
Разработать алгоритм и составить, отладить и выполнить программы с использованием разветвлений и вложенных циклов.

## Задания

### Задание 1

### Условие

### Программный код

  
Рисунок 1 – Условие задания 1

#include <stdio.h>

#include <stdlib.h>

#include <math.h>

#include <locale.h>

int main()

{

float x, a, m, c, f, o, d, g, y;

setlocale (LC\_ALL,"");

printf("Введите х: ");

scanf("%f", &x);

if(x<0){

printf("Введите a: ");

scanf("%f", &a);

if(x<a){

printf("Функция не определена для заданного х.");

}

else{

printf("Введите o: ");

scanf("%f", &o);

y=o\*x/a;

printf("Ваш y=%f ", y);

}

}

else if(x==0){

printf("Ваш y=0");

}

else{

printf("Введите m: ");

scanf("%f", &m);

if(x<m){

printf("Введите f: ");

scanf("%f", &f);

y=f\*x/m;

printf("Ваш y=%f ", y);

}

else{

printf("Введите f: ");

scanf("%f", &f);

printf("Введите c: ");

scanf("%f", &c);

if(x<c){

printf("Введите o: ");

scanf("%f", &o);

y=((x-m)\*(o-f))/(c-m)+f;

printf("Ваш y=%f ", y);

}

else{

printf("Введите d: ");

scanf("%f", &d);

printf("Введите o: ");

scanf("%f", &o);

if(x<=d){

printf("Введите g: ");

scanf("%f", &g);

y=((x-c)\*(g-o))/(d-c)+o;

printf("Ваш y=%f ", y);

}

else{

printf("Функция не определена для заданного x.");

}

}

}

}

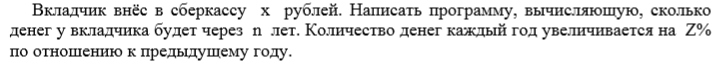
return 0;

}

  
Рисунок 2 – Решение задания 1

### Задание 2

### Условие

  
Рисунок 4 – Условие задания 2

### Блок-схема алгоритма

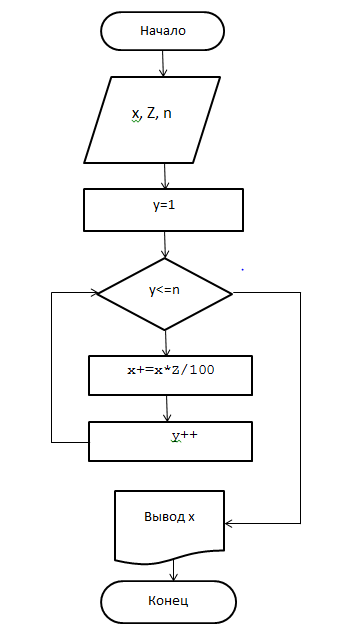


Рисунок 3 – Блок-схема к заданию 2

### Блок-схема алгоритма

### Программный код

#include <stdio.h>

#include <stdlib.h>

#include <math.h>

#include <locale.h>

int main()

{float x,Z, n, y=1;

setlocale(LC\_ALL,"");

printf("Введите количество денег(руб):");

scanf("%f",&x);

printf("На сколько времени вы ложите деньги в банк(лет)?:");

scanf("%f",&n);

printf("Cколько процентов годовых?:");

scanf("%f",&Z);

while (y<=n){

x+=x\*Z/100;

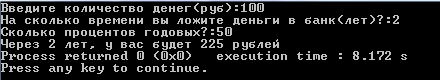
y++;

}

printf("Через %f лет, у вас будет %f рублей ",n,x );

return 0;}

### Выводы по работе

  
Рисунок 4 – Решение задания 2

Разработал алгоритмы и составил, отладил и выполнил программы с использованием разветвлений и вложенных циклов.