Министерство цифрового развития, связи и массовых коммуникаций Российской Федерации

Ордена Трудового Красного Знамени федеральное государственное

бюджетное образовательное учреждение

высшего образования

«Московский технический университет связи и информатики»

Кафедра «Информационная безопасность»

Лабораторная №3

Выполнила:

студентка группы БП31801

Зелик М.Б.

Вариант №10

Москва, 2021

**1. Общее задание**

**Задание**

По номеру Вашего варианта выбрать задачу, решаемую в этой лабораторной работе, и выполнить для нее следующие задания.

***Задание 1***

Составить программу, имеющую линейный алгоритм и состоящую из одной функции void main(). Программу записать в файл с именем task1.c. Скомпилировать, скомпоновать и выполнить.

В функции main организовать вычисление задачи вашего варианта дважды:

* для исходных данных, значения которых задать в виде констант в тексте функции main;
* для исходных произвольных данных, значения которых пользователь Вашей программы должен ввести с клавиатуры в процессе выполнения программы.

Проанализировать результаты работы и сделать выводы. Перенести полученные результаты в отчет.

Вывод действительных чисел осуществлять с точностью до 0.0001

Примеры вывода:

|  |  |
| --- | --- |
| Для функции одной переменной | Для функции двух переменных |
| x = 5  y = 8.2537  f = 0.4514  x = 3.0051  y = 21  f = 84.5201 | x = 5  f = 0.4514  x = 3.0051  f = 84.5201 |

***Задание 2***

Линейный алгоритм функции из задания 1 разделить на две процедуры, выделив в одну вычислительные операции этого алгоритма, а в другую все операции ввода-вывода. Каждую процедуру оформить как функцию. Вычислительную часть алгоритма оформить как функцию с параметрами, передаваемыми по значению, и возвращаемым значением. Прототип функции:

double f(double x) // Если функция имеет один параметр

double f(double x, double y) // Если функция имеет два параметра

Другую часть алгоритма оформить как функцию void main(), вызывающую первую функцию нужное количество раз. Записать тексты функций файл с именем task2.с в следующем порядке: функция с параметрами, функция main. Cкомпилировать, скомпоновать и выполнить.

***Задание 3***

Создать файл task3.с, в котором изменить порядок записи текстов функций, созданных в задании 2. Функции записать в следующем порядке: функция main, функция с параметрами, организующая вычисления (double f(double x) или double f(double x, double y)). Внести требуемые дополнения, добиться успешной компиляции, скомпоновать и выполнить.

***Задание 4***

По-прежнему линейный алгоритм функции из задания 1 разделить на две процедуры, выделив в одну вычислительные операции этого алгоритма, а в другую все операции ввода-вывода.

Вычислительную часть алгоритма оформить как функцию без параметров и без возвращаемого значения. Прототип функции void f(void)

Другую часть алгоритма оформить как функцию void main(), вызывающую вычислительную функцию. Обмен данными между функциями организовать через глобальные объекты (double x, y – аргументы, double result – результат вычисления). Записать тексты функций в файл с именем task4.с в следующем порядке: функция main, затем функция без возвращаемого значения и без параметров. Cкомпилировать, скомпоновать и выполнить.

***Задание 5***

В этом задании необходимо разделить текст файла task3.с на два файла. В первый файл с именем task5\_main.c поместить текст функции main. Скомпилировать только файл task5\_main.c.

Во второй файл с именем task5\_func.с поместить текст функции с параметрами (double f(double x) или double f(double x, double y)). Скомпилировать только файл task5\_func.c. После раздельной компиляции осуществить совместную компоновку. Полученный исполняемый файл выполнить.

Проанализировать результаты работы и сделать выводы.

***Задание 6***

В этом задании необходимо разделить текст файла task4.с на два файла.

В первый файл с именем task6\_main.c поместить текст функции main. Скомпилировать только файл task6\_main.c.

Во второй файл с именем task6\_func.с поместить определения глобальных объектов (double x, y – аргументы, double result – результат вычисления) и текст функции без параметров (void f(void)). Скомпилировать только файл task6\_func.c.

После раздельной компиляции осуществить совместную компоновку. Разобраться в проблемах, возникающих при совместной компиляции и компоновки. Полученный исполняемый файл выполнить.

Проанализировать результаты работы и сделать выводы.

***Задание 7***

В этом задании необходимо модифицировать тексты файлов из задания 6.

Описание функции и внешних переменных выделить в отдельный заголовочный файл func.h, включить его содержимое в файлы task7\_main.c и task7\_func.c. Определение внешних переменных произвести в файле task7\_func.c.

Скомпилировать по отдельности файлы task7\_main.c и task7\_func.c. Произвести компоновку, выполнить полученный исполняемый файл.

***Задание 8***

Скомпилировать по отдельности файлы из задания 7 task7\_main.c и task7\_func.c. Создать статическую библиотеку task8\_lib.lib, включающую объектный файл task7\_func.obj. Полученную статическую библиотеку скомпоновать с файлом task7\_main.obj. Выполнить полученный исполняемый файл.

**2. Индивидуальное задание**

Вариант №10

**3. Выполнение**

**Задание 1**

#include "stdio.h"

#define \_USE\_MATH\_DEFINES

#include "math.h"

double summ(int n)

{

    double res =0;

    int i = 0;

    do

    {

        res += pow((-1),i)\*(i+1)/(pow(i,3)-pow(i,2)+1);

        ++i;

    } while (i < n);

    return res;

}

Листинг 1. Программный код task1\_do.c

#include "stdio.h"

#define \_USE\_MATH\_DEFINES

#include "math.h"

double summ(int n)

{

    double res = 0;

    for (int i = 0; i < n; ++i)

    {

        res += pow((-1),i)\*(i+1)/(pow(i,3)-pow(i,2)+1);

    }

    return res;

}

Листинг 2. Программный код task1\_for.c

#include "stdio.h"

#define \_USE\_MATH\_DEFINES

#include "math.h"

double summ(int n)

{

    int i = 0;

    double res = 0;

    while ((i < n))

    {

        res += pow((-1),i)\*(i+1)/(pow(i,3)-pow(i,2)+1);

        i++;

    }

    return res;

}

Листинг 3. Программный код task1\_while.c

**Задание 2**

#include "stdio.h"

#define \_USE\_MATH\_DEFINES

#include "math.h"

double summ2(double eps)

{

    int i = 0;

    double res = pow((-1),i)\*(i+1)/(pow(i,3)-pow(i,2)+1);

    do

    {

        ++i;

        res += pow((-1),i)\*(i+1)/(pow(i,3)-pow(i,2)+1);

    } while (fabs(res) >= eps);

    return res;

}

Листинг 4. Программный код task2\_do.c

#include "stdio.h"

#define \_USE\_MATH\_DEFINES

#include "math.h"

double summ2(double eps)

{

    double res = 0;

    for (int i = 0; ++i)

    {

        res += pow((-1),i)\*(i+1)/(pow(i,3)-pow(i,2)+1);

        if (fabs(res) <= eps) break;

    }

    return res;

}

Листинг 5. Программный код task2\_for.c

#include "stdio.h"

#define \_USE\_MATH\_DEFINES

#include "math.h"

double summ2(double eps)

{

    int i = 0;

    double res = res = pow((-1),i)\*(i+1)/(pow(i,3)-pow(i,2)+1);;

    while (fabs(res) >= eps)

    {

        ++i;

        res += pow((-1),i)\*(i+1)/(pow(i,3)-pow(i,2)+1);

    }

    return res;

}

Листинг 6. Программный код task2\_while.c

**Задание 3**

#include "stdio.h"

#define \_USE\_MATH\_DEFINES

#include "math.h"

void print(int n, int k)

{

    double res = 0;

    int i=0;

    do

    {

        res += pow((-1),i)\*(i+1)/(pow(i,3)-pow(i,2)+1);

        ++i;

        if (i % k == 0) continue;

        printf("%lf", &res);

        }

    } while (i < n);

}

Листинг 7. Программный код task3\_do.c

#include "stdio.h"

#define \_USE\_MATH\_DEFINES

#include "math.h"

void print(int n, int k)

{

    int res = 0;

    for (int i = 0; i < n; ++i)

    {

        if (i % k == k-1) continue;

            res += pow((-1),i)\*(i+1)/(pow(i,3)-pow(i,2)+1);

            printf("%lf", &res);

        }

        i++;

    }

}

Листинг 8. Программный код task3\_for.c

#include "stdio.h"

#define \_USE\_MATH\_DEFINES

#include "math.h"

void print(int n, int k)

{

    double res = 0;

    int i=0;

    while (i < n)

    {

        res += pow((-1),i)\*(i+1)/(pow(i,3)-pow(i,2)+1);

        ++i;

        if (i % k == 0) continue;

        printf("%lf", &res);

    }

}

Листинг 9. Программный код task3\_while.c

**Задание 4**

#include "stdio.h"

#define \_USE\_MATH\_DEFINES

#include "math.h"

int findFirstElement(double eps)

{

    int i = 0;

    int a = 0;

    double res = pow((-1),i)\*(i+1)/(pow(i,3)-pow(i,2)+1);

    do

    {

        if (fabs(res) <= eps) break;

        ++i;

        a = i;

        res += pow((-1),i)\*(i+1)/(pow(i,3)-pow(i,2)+1);

        }

    } while (i > 0);

    return a;

}

Листинг 10. Программный код task4\_do.c

#include "stdio.h"

#define \_USE\_MATH\_DEFINES

#include "math.h"

int findFirstElement(double eps)

{

    int a = 0;

    double res = 0;

    for (int i = 0; ++i)

    {

        a = i;

        res += pow((-1),i)\*(i+1)/(pow(i,3)-pow(i,2)+1);

        if (fabs(res) <= eps) break;

    }

    return k;

}

Листинг 11. Программный код task4\_for.c

#include "stdio.h"

#define \_USE\_MATH\_DEFINES

#include "math.h"

int res = 0;

int findFirstElement(double eps)

{

    int i = 0;

    int a = 0;

    res = pow((-1),i)\*(i+1)/(pow(i,3)-pow(i,2)+1);

    while (i > 0)

    {

        if (fabs(res) <= eps) break;

        ++i;

        a = i;

        res += pow((-1),i)\*(i+1)/(pow(i,3)-pow(i,2)+1);

    }

            return a;

}

Листинг 12. Программный код task4\_while.c

**Задание 5**

#include "stdio.h"

#define \_USE\_MATH\_DEFINES

#include "math.h"

int findFirstNegativeElement(double eps)

{

    double res = pow((-1),i)\*(i+1)/(pow(i,3)-pow(i,2)+1);

    int i = 0;

    int a = 0;

    do

    {

        ++i;

        res += pow((-1),i)\*(i+1)/(pow(i,3)-pow(i,2)+1);

        if (res > 0) continue;

        a = i;

        if (fabs(res) <= eps) return a;

    } while (i > 0);

}

Листинг 13. Программный код task5\_do.c

#include "stdio.h"

#define \_USE\_MATH\_DEFINES

#include "math.h"

double findFirstNegativeElement(double eps)

{

    double res = 0;

    int a;

    for (int i = 0; ++i)

    {

        a = i;

        res += pow((-1),i)\*(i+1)/(pow(i,3)-pow(i,2)+1);

        if (res > 0) continue;

        if (fabs(res) <= eps) return k;

    }

}

Листинг 14. Программный код task5\_for.c

#include "stdio.h"

#define \_USE\_MATH\_DEFINES

#include "math.h"

double findFirstNegativeElement(double eps)

{

    double res = pow((-1),i)\*(i+1)/(pow(i,3)-pow(i,2)+1);

    int i = 0;

    int a = 0;

    while (i > 0)

    {

        ++i;

        res += pow((-1),i)\*(i+1)/(pow(i,3)-pow(i,2)+1);

        if (res > 0) continue;

        a = i;

        if (fabs(res) <= eps) return k;

    }

}

Листинг 15. Программный код task5\_while.c

**Задание 6**

#include "stdio.h"

#include "stdbool.h"

#include <conio.h>

#include "stdlib.h"

int findFirstNegativeElement(double eps);

int findFirstElement(double eps);

void print(int n, int k);

double summ2(double eps);

double summ(int n);

double result;

double eps;

int n, k;

int main()

{

  printf("1. Zadanie 1\n2. Zadanie 2\n3. Zadanie 3\n4. Zadanie 4\n5. Zadanie 5\n6. Exit\n");

  switch (\_getch())

  {

    case '1':

      printf("Zadanie 1\n Vvedite n: ");

      scanf("%lf", &n);

      result = summ(n);

      printf("%lf\n", result);

        break;

    case '2':

      printf("Zadanie 2\nVvedite E ");

      scanf("%lf", &eps);

      result = summ2(eps);

      printf("%lf\n", result);

      break;

      case '3':

      printf("Zadanie 3\n Vvedite n k: ");

      scanf("%d %d", &n, &k);

      print(n, k);

         break;

        case '4':

      printf("Zadanie 4\n Vvedite E: ");

      scanf("%lf", &eps);

      result = findFirstElement(eps);

       printf("%d\n", result);

        break;

        case '5':

      printf("Zadanie 5\n Vvedite E: ");

      scanf("%lf", &eps);

      result = findFirstNegativeElement(eps);

      printf("%d\n", result);

        break;

        case '6':

      exit(1);

        break;

  }

  system("pause");

  return 0;

}

Листинг 16. Программный код lab3.c

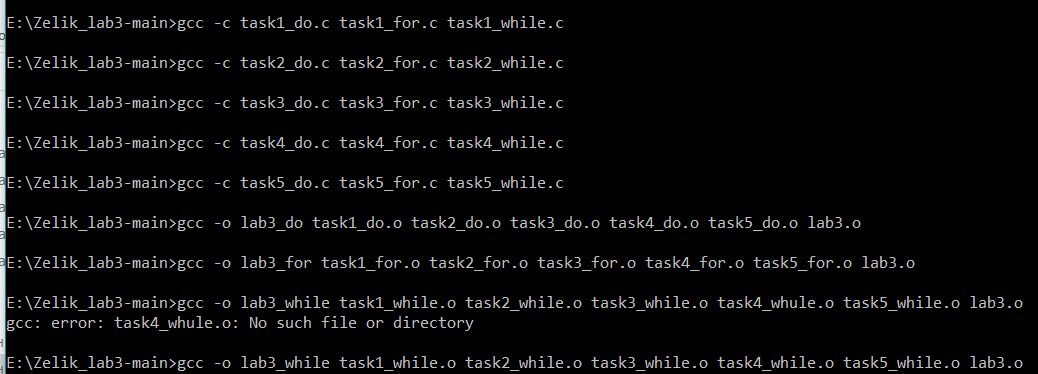


Рис 1. Сборка lab3\_do, lab3\_for, lab3\_while