**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»**

**(МТУСИ)**

Кафедра

«Сети и системы связи»

ЛАБОРАТОРНАЯ РАБОТА №1

по дисциплине:

«ПвСИБ»

на тему:

«Компиляция файлов исходного кода и компоновка полученных объектных файлов в исполняемый модуль. Линейные алгоритмы»

Выполнил:

студент 4 курса

группы БСУ1801

Неронов Ф.А.

Вариант №17

Проверил:

Барков В.В.

Москва 2021

1. **Цель работы:**

Овладеть навыками создания однофайловых и многофайловых проектов в интегрированной среде разработки Microsoft Visual Studio 2015 Community Edition и научится создавать линейные программы на языке C с применением арифметических операций.

1. **Общее задание.**

#### **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

#### **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.

1. **Индивидуальное задание**
2. **Результаты**

Для исходных данных, значения которых задать в виде констант в тексте функции main:

x = 9.0000

f = 3.0000

Для исходных произвольных данных, значения которых пользователь программы вводит с клавиатуры в процессе выполнения программы:

x = 16.0000

f = 4.0000

1. **Выполнение**

**Выполнение задания 1:**

Листинг 1 - Исходный код файла task1.c

#include <stdio.h>

#include <math.h>

void main(void)

{

double x = 9;

double f = sqrt(pow(x \* 3 + 2, 2) - 24 \* x) / (3 \* sqrt(x) - (2 / sqrt(x)));

printf("x = %.4lf\n", x);

printf("f = %.4lf\n\n", f);

printf("x = ");

scanf("%lf", &x);

f = sqrt(pow(x \* 3 + 2, 2) - 24 \* x) / (3 \* sqrt(x) - (2 / sqrt(x)));

printf("f = %.4lf", f);

}

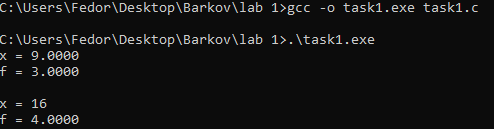


Рис.1.1 – Результат компиляции и выполнения task1.exe

**Выполнение задания 2:**

Листинг 2 - Исходный код файла task2.c

#include <stdio.h>

#include <math.h>

double f(double x)

{

return sqrt(pow(x \* 3 + 2, 2) - 24 \* x) / (3 \* sqrt(x) - (2 / sqrt(x)));

}

void main(void)

{

double x = 9;

printf("x = %.4lf\n", x);

printf("f = %.4lf\n\n", f(x));

printf("x = ");

scanf("%lf", &x);

printf("f = %.4lf", f(x));

}

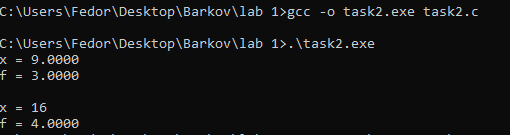


Рис.1.2 – Результат компиляции и выполнения task2.exe

**Выполнение задания 3:**

Листинг 3 - Исходный код файла task3.c

#include <stdio.h>

#include <math.h>

double f(double x);

void main(void)

{

double x = 9;

printf("x = %.4lf\n", x);

printf("f = %.4lf\n\n", f(x));

printf("x = ");

scanf("%lf", &x);

printf("f = %.4lf", f(x));

}

double f(double x)

{

double f;

f = sqrt(pow(x \* 3 + 2, 2) - 24 \* x) / (3 \* sqrt(x) - (2 / sqrt(x)));

return(f);

}

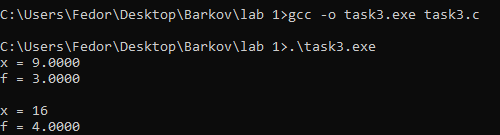


Рис.1.3 – Результат компиляции и выполнения task3.exe

**Выполнение задания 4:**

Листинг 4 - Исходный код файла task4.c

#include <stdio.h>

#include <math.h>

void f(void);

double x, y;

void main(void)

{

x = 9;

f();

printf("x = %.4lf\n", x);

printf("f = %.4lf\n\n", y);

printf("x =");

scanf("%lf", &x);

f();

printf("f = %.4lf", y);

}

void f(void)

{

y = sqrt(pow(x \* 3 + 2, 2) - 24 \* x) / (3 \* sqrt(x) - (2 / sqrt(x)));

}

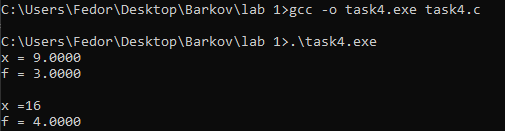


Рис.1.4 – Результат компиляции и выполнения task4.exe

**Выполнение задания 5:**

Листинг 5 - Исходный код файла task5\_main.c

#include <stdio.h>

double f(double x);

void main(void)

{

double x = 9;

printf("x = %.4lf\n", x);

printf("f = %.4lf\n\n", f(x));

printf("x = ");

scanf("%lf", &x);

printf("f=%.4lf", f(x));

}

Листинг 6 - Исходный код файла task5\_func.c

#include <stdio.h>

#include <math.h>

double f(double x)

{

double f;

return sqrt(pow(x \* 3 + 2, 2) - 24 \* x) / (3 \* sqrt(x) - (2 / sqrt(x)));

}

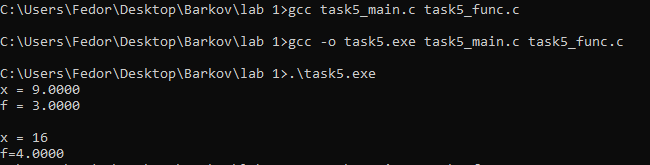


Рис.1.5 – Результат компиляции, компоновки и выполнения task5.exe

**Выполнение задания 6:**

Листинг 7 - Исходный код файла task6\_main.c

#include <stdio.h>

double x, y;

void f(void);

void main(void)

{

x = 9;

f();

printf("x = %.4lf\n", x);

printf("f = %.4lf\n\n", y);

printf("x = ");

scanf("%lf", &x);

f();

printf("f = %.4lf", y);

}

Листинг 8 - Исходный код файла task6\_func.c

#include <stdio.h>

#include <math.h>

extern double x, y;

void f(void)

{

y = sqrt(pow(x \* 3 + 2, 2) - 24 \* x) / (3 \* sqrt(x) - (2 / sqrt(x)));

}

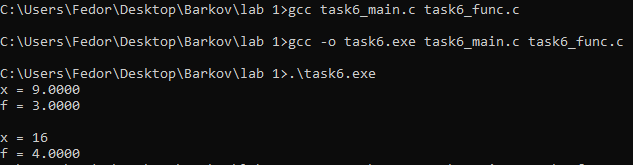


Рис.1.6 – Результат компиляции, компоновки и выполнения task6.exe

**Выполнение задания 7:**

Листинг 9 - Исходный код файла task7\_main.c

#include <stdio.h>

#include <math.h>

#include "func.h"

void main(void)

{

x = 9;

f();

printf("x = %.4lf\n", x);

printf("f = %.4lf\n\n", y);

printf("x = ");

scanf("%lf", &x);

f();

printf("f = %.4lf", y);

}

Листинг 10 - Исходный код файла task7\_func.c

#include <math.h>

#include "func.h"

double x, y;

void f(void)

{

y = sqrt(pow(x \* 3 + 2, 2) - 24 \* x) / (3 \* sqrt(x) - (2 / sqrt(x)));

}

Листинг 11 - Исходный код файла func.h

#pragma once

extern double x, y;

void f();

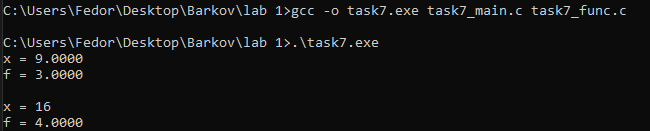


Рис.1.7 – Результат компиляции, компоновки и выполнения task7.exe

**Выполнение задания 8:**

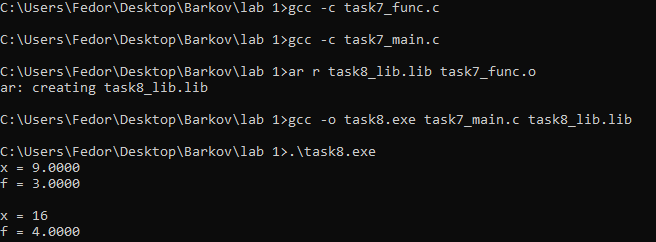


Рис.1.8 – Результат компиляции, компоновки и выполнения task8.exe

Ссылка на github

<https://github.com/BSU1801/Barkov.git>

1. **Вывод:**

На этой лабораторной работе мы овладели навыками создания однофайловых и многофайловых проектов в интегрированной среде разработки Microsoft Visual Studio 2015 Community Edition и научится создавать линейные программы на языке C с применением арифметических операций.