**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 «Информационная безопасность»

**ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №2**

по дисциплине

**«Разработка безопасного ПО»**

на тему

«Компиляция файлов исходного кода и компоновка полученных объектных файлов в исполняемый модуль. Объявление и определение переменных, функций и пространств имён. Линейные алгоритмы»

**Вариант №14**

Выполнил:

студент группы БПЗ1902

Оглоблин Д.И.

Проверил:

старший преподаватель кафедры ИБ

Барков В.В.

Москва, 2021

# 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.

# 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.cpp. Скомпилировать, скомпоновать и выполнить.

В функции main организовать вычисление задачи вашего варианта дважды:

* для исходных данных, значения которых задать в виде литералов 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.

Вычислительную часть алгоритма оформить в виде функции f.

Входные данные необходимо передавать через **параметры функции по значению**.

Выходные данные необходимо возвращать через **возвращаемое значение функции**.

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 функцию f нужное количество раз.

Записать тексты функций файл с именем task2.сpp в следующем порядке: функция f, функция main.

Cкомпилировать, скомпоновать и выполнить.

#### Задание 3

Создать файл task3.сpp, в котором изменить порядок записи текстов функций, созданных в задании 2. Функции записать в следующем порядке: функция main, функция f.

Внести требуемые дополнения, добиться успешной компиляции, скомпоновать и выполнить.

#### Задание 4

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 в виде функции f.

Входные данные необходимо передавать через **параметры функции по ссылке на константный объект**.

Выходные данные необходимо возвращать через **дополнительный параметр функции по ссылке**. Функция не имеет возвращаемого значения.

Прототип функции:

void f(const double& x, double& result) // или

void f(const double& x, const double& y, double& result)

Другую часть алгоритма оформить как функцию void main(), вызывающую функцию f нужное количество раз.

Записать тексты функций в файл с именем task4.сpp в следующем порядке: функция main, затем функция f. Cкомпилировать, скомпоновать и выполнить.

#### Задание 5

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 в виде функции f.

Входные данные необходимо передавать через **глобальные объекты x и y**.

Выходные данные необходимо возвращать через **глобальный объект result**. Функция не имеет возвращаемого значения.

Прототип функции:

void f();

Другую часть алгоритма оформить как функцию void main(), вызывающую функцию f нужное количество раз.

Записать тексты функций в файл с именем task5.сpp в следующем порядке: функция main, затем функция f. Cкомпилировать, скомпоновать и выполнить.

#### Задание 6

В этом задании необходимо разделить текст файла task2.сpp на два файла. В первый файл с именем task6\_main.cpp поместить текст функции main. Скомпилировать только файл task6\_main.cpp.

Во второй файл с именем task6\_func.сpp поместить текст функции f. Скомпилировать только файл task6\_func.cpp.

После раздельной компиляции осуществить совместную компоновку. Полученный исполняемый файл выполнить.

Проанализировать результаты работы и сделать выводы.

#### Задание 7

В этом задании необходимо разделить текст файла task5.сpp на два файла.

В первый файл с именем task7\_main.cpp поместить текст функции main. Скомпилировать только файл task7\_main.cpp.

Во второй файл с именем task7\_func.сpp поместить определения глобальных объектов x, y и result и текст функции f.

Скомпилировать только файл task7\_func.cpp.

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 8

В этом задании необходимо модифицировать тексты файлов из задания 7.

Описание функции и внешних переменных выделить в отдельный заголовочный файл func.h, включить его содержимое в файлы task8\_main.cpp и task8\_func.cpp. Определение внешних переменных произвести в файле task8\_func.cpp.

Скомпилировать по отдельности файлы task8\_main.cpp и task8\_func.cpp. Произвести компоновку, выполнить полученный исполняемый файл.

#### Задание 9

В этом задании необходимо модифицировать тексты файлов из задания 8.

Перенести все объявления и определения функции f в пространство имен, содержащее Вашу фамилию.

Скомпилировать по отдельности файлы task9\_main.cpp и task9\_func.cpp. Произвести компоновку, выполнить полученный исполняемый файл.

#### Задание 10

В этом задании необходимо собрать статическую библиотеку task10\_lib.lib. В библиотеку включить объектный файл task9\_func.obj из задания 9.

Произвести компоновку объектного файла task9\_main.obj из задания 9 и полученную статическую библиотеку task10\_lib.lib. Выполнить полученный исполняемый файл.

# Индивидуальный вариант задания

# Выполнение домашнего задания

*Ссылка на гитхаб:* https://github.com/n3rfy/labs\_cpp/tree/main/lab2

**Задание 1**

**Код task1.cpp:**

#include <iostream>

#include <cmath>

using namespace std;

int main() {

double x;

double result;

while(true){

cout << "Enter x: ";

cin >> x;

result = (cos(x) + sin(x)) / (cos(x) - sin(x));

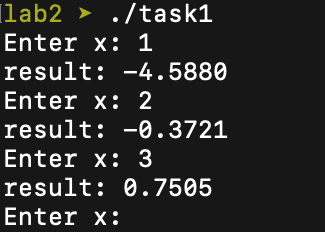
printf("result: %.4f\n", result);

}

return 0;

}

**Выполнение:**

****

**Задание 2**

**Код task2.cpp:**

#include <iostream>

#include <cmath>

using namespace std;

double f(double x){

return (cos(x) + sin(x)) / (cos(x) - sin(x));

}

int main(){

double x;

while(true){

cout << "Enter x: ";

cin >> x;

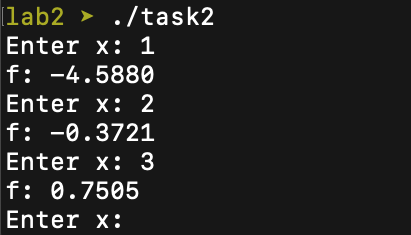
printf("f: %.4f\n", f(x));

}

return 0;

}

**Выполнение:**

****

**Задание 3**

**Код task3.cpp:**

#include <iostream>

#include <cmath>

using namespace std;

double f(double);

int main(){

double x;

while (true){

cout << "Enter x: ";

cin >> x;

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

}

return 0;

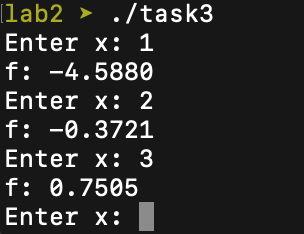
}

double f(double x){

return (cos(x) + sin(x)) / (cos(x) - sin(x));

}

**Выполнение:**

****

**Задание 4**

**Код task4.cpp:**

#include <iostream>

#include <cmath>

using namespace std;

void f(const double&, double&);

int main(){

double x;

double result;

while(true){

cout << "Enter x: ";

cin >> x;

f(x,result);

printf("f: %.4f\n", result);

}

return 0;

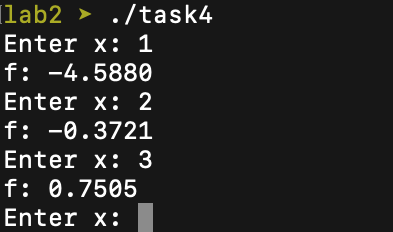
}

void f(const double& x, double& result){

result = (cos(x) + sin(x)) / (cos(x) - sin(x));

}

**Выполнение:**

****

**Задание 5**

**Код task5.cpp:**

#include <iostream>

#include <cmath>

using namespace std;

double x;

double result;

void f();

int main() {

while(true){

cout << "Enter x: ";

cin >> x;

f();

printf("f: %.4f\n", result);

}

return 0;

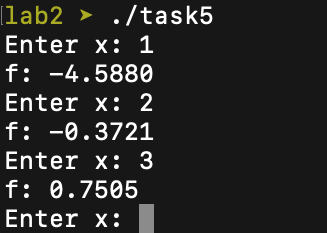
}

void f() {

result = (cos(x) + sin(x)) / (cos(x) - sin(x));

}

**Выполнение:**

****

**Задание 6**

**Код task6\_main.cpp:**

#include <iostream>

using namespace std;

double f(double);

int main(){

double x;

while(true){

cout << "Enter x: ";

cin >> x;

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

}

return 0;

}

**Код task6\_func.cpp:**

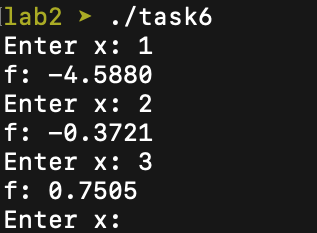
#include <cmath>

double f(double x) {

return (cos(x) + sin(x)) / (cos(x) - sin(x));

}

**Выполнение:**

****

**Задание 7**

**Код task7\_main.cpp:**

#include <iostream>

using namespace std;

extern double x, result;

extern void f();

int main(){

while(true){

cout << "Enter x: ";

cin >> x;

f();

printf("f: %.4f\n", result);

}

return 0;

}

**Код task7\_func.cpp:**

#include <cmath>

double x;

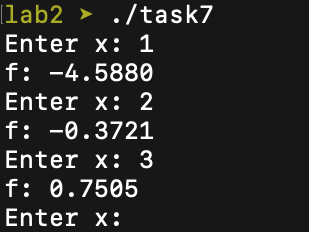
double result;

void f() {

result = (cos(x) + sin(x)) / (cos(x) - sin(x));

}

**Выполнение:**

****

**Задание 8**

**Код task8\_main.cpp:**

#include "func.h"

int main(){

while(true){

cout << "Enter x: ";

cin >> x;

f();

printf("f: %.4f\n", result);

}

return 0;

}

**Код task8\_func.cpp:**

#include "func.h"

double x, result;

void f() {

result = (cos(x) + sin(x)) / (cos(x) - sin(x));

}

**Код func.h:**

#include <iostream>

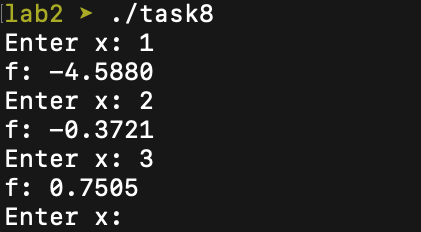
#include <cmath>

using namespace std;

extern double x, result;

extern void f();

**Выполнение:**

****

**Задание 9**

**Код task9\_main.cpp:**

#include "ogloblin.h"

using namespace std;

int main(){

while(true){

cout << "Enter x: ";

cin >> ogloblin::x;

ogloblin::f();

printf("f: %.4f\n", ogloblin::result);

}

return 0;

}

**Код task9\_func.cpp:**

#include "ogloblin.h"

namespace ogloblin{

double x, result;

void f() {

result = (cos(x) + sin(x)) / (cos(x) - sin(x));

}

}

**Код zhdanov.h:**

#include <iostream>

#include <cmath>

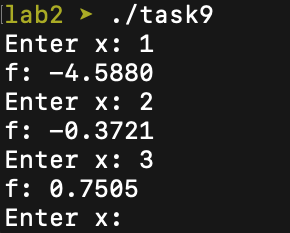
namespace ogloblin{

extern double x, result;

void f();

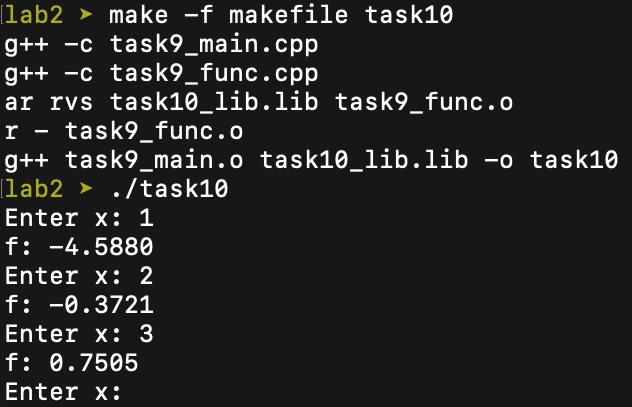
}

**Выполнение:**

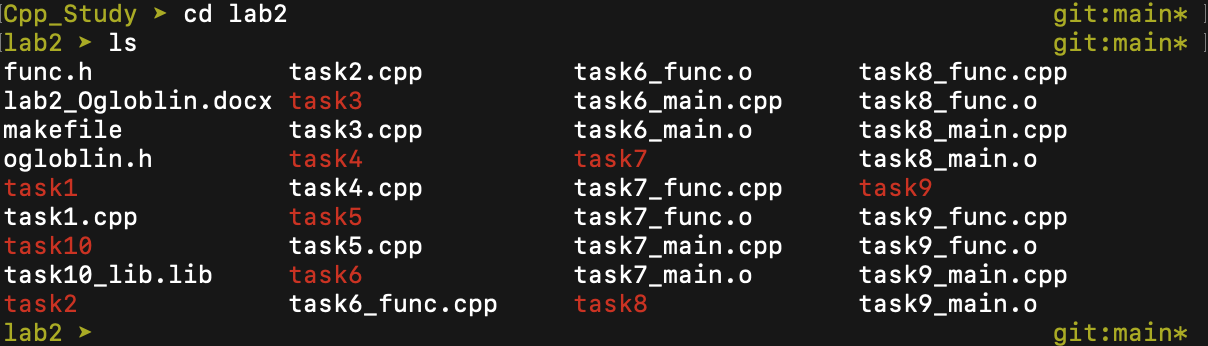
****

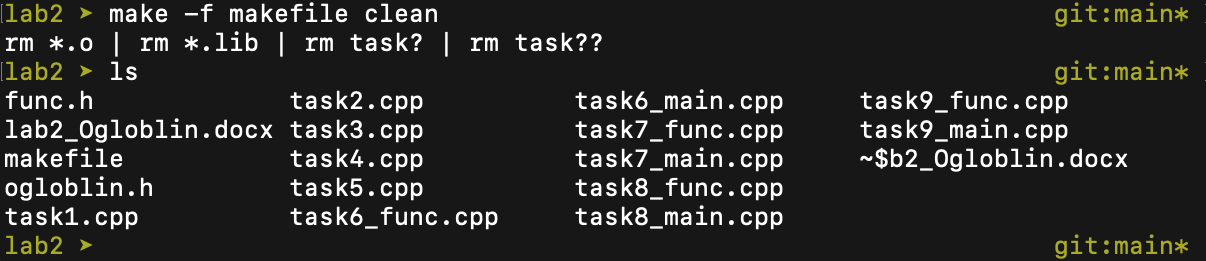
**Задание 10**

**Выполнение:**

****

**Очистка:**

****

****

**Код makefile’a:**

task1:

g++ $@.cpp -o $@

task2:

g++ $@.cpp -o $@

task3:

g++ $@.cpp -o $@

task4:

g++ $@.cpp -o $@

task5:

g++ $@.cpp -o $@

task6:

g++ -c $@\_main.cpp

g++ -c $@\_func.cpp

g++ $@\_main.o $@\_func.o -o $@

task7:

g++ -c $@\_main.cpp

g++ -c $@\_func.cpp

g++ $@\_main.o $@\_func.o -o $@

task8:

g++ -c $@\_main.cpp

g++ -c $@\_func.cpp

g++ $@\_main.o $@\_func.o -o $@

task9:

g++ -c $@\_main.cpp

g++ -c $@\_func.cpp

g++ $@\_main.o $@\_func.o -o $@

task10:

g++ -c task9\_main.cpp

g++ -c task9\_func.cpp

ar rvs $@\_lib.lib task9\_func.o

g++ task9\_main.o task10\_lib.lib -o $@

clean:

rm \*.o | rm \*.lib | rm task? | rm task??