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 №3

по дисциплине

«Разработка безопасного ПО»

Вариант №16

Выполнил:

студент группы БПЗ1901

Неживлева К.И.

Проверил:

старший преподаватель кафедры ИБ

Барков В.В.

Москва, 2021

**Цель практикума**

Овладеть навыками создания модулей в языке C++ и научится создавать линейные, разветвляющиеся и итерационные программы на языке C++.

**Задание**

По номеру Вашего варианта выбрать функции , , .

В каждом задании необходимо реализовать функции:

, , , и

При реализации функций и вычисление очередного слагаемого организовать с помощью функции .

Вычисление значения фукнции следует производить до тех пор, пока не будет достигнута заданная точность , т.е. до тех пор, пока истинно условие .

Разработанные фукнции разместить в отдельном модуле. Имя модуля должно состоять из названия группы, фамилии исполнителя, номера лабораторной работы и номера задания. Все части названия модуля разделить точкой. Например, для задания 1: BPZ1901.Ivanov.Lab3.Task1.

В модуле определить пространство имен с названием дисциплины (RBPO). Внутри пространства имен RBPO определить пространство имен, включающее номер лабораторной работы (Lab3). Внутри пространства имен RBPO::Lab3 определить пространство имен с номером задания (TaskN, где N – номер задания).

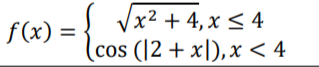
Все разработанные функции в задании N должны лежать в пространстве имен RBPO::Lab3::TaskN, где N – номер задания.

**Индивидуальное задание**

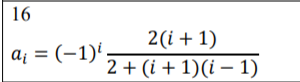
Функция f(x):

# 

Функция f2(x):



Функция a(n):



Задание 1

При реализации функции использовать условное выражение (не инструкцию выбора if).

Функции и реализовать с помощью цикла for.

Определения функций , , , , поместить в одну единицу трансляции, описывающую интерфейс модуля (Module Interface Unit – файл с расширением \*.ixx).

Экспортировать функции , , , . Само пространство имен НЕ экспортировать.

При необходимости подключить заголовочные файлы, используйте фрагмент глобального модуля (Global Module Fragment).

Листинг 1 – BPZ1901. Nezhivleva.Lab3.Task1.ixx

export module BPZ1901.Nezhivleva.Lab3.Task1;

import <cmath>;

namespace RBPO {

namespace Lab3 {

namespace Task1 {

export double f1(double x);

export double f2(double x);

export double a(int i);

export double f3(int n);

export double f4(double eps);

}

}

}

Листинг 2 – BPZ1901. Nezhivleva.Lab3.Task1.cpp

module BPZ1901.Nezhivleva.Lab3.Task1;

namespace RBPO {

namespace Lab3 {

namespace Task1 {

double f1(double x) {

return (pow(x, 2.0) + 2.0 \* x - 3 + (x + 1) \* sqrt(pow(x, 2) - 9)) / (pow(x, 2.0) - 2.0 \* x - 3 + (x + 1) \* sqrt(pow(x, 2) - 9));

}

double f2(double x) {

return (x <= 4.0) ? (sqrt(pow(x, 2.0) + 4.0)) : (cos(abs(2.0 + x)));

}

double a(int i) {

return ((2 \* (i + 1)) / (2 + (i + 1) \* (1 - 1))) \* ((i % 2) ? -1.0 : 1.0);

}

double f3(int n) {

double sum = 0.0;

for (int i = 0; i <= n; i++) {

sum += a(i);

}

return sum;

}

double f4(double eps) {

double pred = a(0);

double sled = a(1);

double sum = pred+sled;

for (int i = 2; abs(a(i) - a(i - 1)) > eps; i++) {

sum += a(i);

}

return sum;

}

}

}

}

Задание 2

При реализации функции использовать инструкцию выбора if.

Функции и реализовать с помощью цикла while.

Объявления функций , , , поместить в единицу трансляции, описывающую интерфейс модуля (Module Interface Unit). Объявление функции помещать не нужно.

Экспортировать пространство имен RBPO::Lab3::Task2 целиком.

Определения функций , f , , поместить в одну единицу трансляции, описывающую реализацию модуля (Module Implementation Unit).

При необходимости подключить заголовочные файлы, используйте используйте фрагмент глобального модуля (Global Module Fragment).

Листинг 3 – BPZ1901. Nezhivleva.Lab3.Task2.cpp

module BPZ1901.Nezhivleva.Lab3.Task2;

namespace RBPO {

namespace Lab3 {

namespace Task2 {

double f1(double x) {

return (pow(x, 2.0) + 2.0 \* x - 3 + (x + 1) \* sqrt(pow(x, 2) - 9)) / (pow(x, 2.0) - 2.0 \* x - 3 + (x + 1) \* sqrt(pow(x, 2) - 9));

}

double f2(double x) {

if (x <= 4.0) return (sqrt(pow(x, 2.0) + 4.0));

else return cos(abs(2.0 + x));

}

double a(int i) {

return ((2 \* (i + 1)) / (2 + (i + 1) \* (1 - 1))) \* ((i % 2) ? -1.0 : 1.0);

}

double f3(int n) {

double sum = 0.0;

int i = 0;

while (i <= n) {

sum += a(i);

i++;

}

return sum;

}

double f4(double eps) {

double pred = a(0);

double sled = a(1);

double sum = pred + sled;

int i = 2;

while (abs(a(i) - a(i - 1)) > eps) {

sum += a(i);

i++;

}

return sum;

}

}

}

}

Листинг 4 – BPZ1901. Nezhivleva.Lab3.Task2.ixx

export module BPZ1901.Nezhivleva.Lab3.Task2;

import <cmath>;

namespace RBPO {

namespace Lab3 {

namespace Task2 {

export double f1(double x);

export double f2(double x);

export double a(int i);

export double f3(int n);

export double f4(double eps);

}

}

}

Задание 3

Функции и реализовать с помощью цикла do … while. Реализации функций должны возвращать те же результаты, что и в предыдущих заданиях.

Объявления функций , , , , поместить в единицу трансляции, описывающую интерфейс модуля (Module Interface Unit).

Экспортировать функции , , , . Само пространство имен НЕ экспортировать.

Каждое определение функций , f , , поместить в отдельную единицу трансляции, описывающую реализацию модуля (Module Implementation Unit).

Итоговый модуль будет содержать одну единицу трансляции, описывающую интерфейс модуля и 5 единиц трансляции, описывающих реализацию модуля. В каждой единице трансляции будет размещено одно определение функции.

При необходимости подключить заголовочные файлы, используйте фрагмент глобального модуля (Global Module Fragment).

Листинг 5 – файл BPZ1901. Nezhivleva.Lab3.Task3.a.cpp

module BPZ1901.Nezhivleva.Lab3.Task3;

namespace RBPO {

namespace Lab3 {

namespace Task3 {

double a(int i) {

return ((2 \* (i + 1)) / (2 + (i + 1) \* (1 - 1))) \* ((i % 2) ? -1.0 : 1.0);

}

}

}

}

Листинг 6 – файл BPZ1901. Nezhivleva.Lab3.Task3.f1.cpp

module BPZ1901.Nezhivleva.Lab3.Task3;

namespace RBPO {

namespace Lab3 {

namespace Task3 {

double f1(double x) {

return (pow(x, 2.0) + 2.0 \* x - 3 + (x + 1) \* sqrt(pow(x, 2) - 9)) / (pow(x, 2.0) - 2.0 \* x - 3 + (x + 1) \* sqrt(pow(x, 2) - 9));

}

}

}

}

Листинг 7 – файл BPZ1901. Nezhivleva.Lab3.Task3.f2.cpp

module BPZ1901.Nezhivleva.Lab3.Task3;

namespace RBPO {

namespace Lab3 {

namespace Task3 {

double f2(double x) {

if (x <= 4.0) return (sqrt(pow(x, 2.0) + 4.0));

else return cos(abs(2.0 + x));

}

}

}

}

Листинг 8 – файл BPZ1901. Nezhivleva.Lab3.Task3.f3.cpp

module BPZ1901.Nezhivleva.Lab3.Task3;

namespace RBPO {

namespace Lab3 {

namespace Task3 {

double f3(int n) {

double sum = 0.0;

int i = 0;

do {

sum += a(i);

i++;

} while (i <= n);

return sum;

}

}

}

}

Листинг 9 – файл BPZ1901.Nezhivleva.Lab3.Task3.f4.cpp

module BPZ1901.Nezhivleva.Lab3.Task3;

namespace RBPO {

namespace Lab3 {

namespace Task3 {

double f4(double eps) {

double pred = a(0);

double sled = a(1);

double sum = pred + sled;

int i = 2;

do {

sum += a(i);

i++;

} while (abs(a(i) - a(i - 1)) > eps);

return sum;

}

}

}

}

Листинг 10 – файл BPZ1901. Nezhivleva.Lab3.Task3.ixx

export module BPZ1901.Nezhivleva.Lab3.Task3;

import <cmath>;

namespace RBPO {

namespace Lab3 {

namespace Task3 {

export double f1(double x);

export double f2(double x);

export double a(int i);

export double f3(int n);

export double f4(double eps);

}

}

}

Задание 4

Модуль разделить на 5 разделов (module partition).

Объявление каждой из функций , , , , поместить в отдельную единицу трансляции, описывающую интерфейс раздела модуля (Module Partition Interface Unit).

Экспортировать функции , , , в соответствующих единицах трасляции.

В единице трансляции, описывающей интерфейс модуля (Module Interface Unit) импортировать разделы, содержащие экспорт фукнций , , , и экспортировать их.

Каждое определение функций , f , , поместить в отдельную единицу трансляции, описывающую реализацию раздела модуля (Module Partition Implementation Unit). В единицах трансляции, содержащих реализации и потребуется импортировать раздел с объявлением функции .

Итоговый модуль будет содержать одну единицу трансляции, описывающую интерфейс модуля и 5 разделов. Каждый из разделов будет включать единицу трансляции, описывающую интерфейс раздела модуля (в которой будет содержаться прототип функции), и единицу трансляции, описывающую реализацию раздела модуля (в которой будет содержаться реализация фукнции).

При необходимости подключить заголовочные файлы, используйте фрагмент глобального модуля (Global Module Fragment).

Листинг 11 – файл BPZ1901. Nezhivleva.Lab3.Task4.a.cpp

module BPZ1901.Nezhivleva.Lab3.Task4:a;

namespace RBPO {

namespace Lab3 {

namespace Task4 {

double a(int i) {

return ((2 \* (i + 1)) / (2 + (i + 1) \* (1 - 1))) \* ((i % 2) ? -1.0 : 1.0);

}

}

}

}

Листинг 12 – файл BPZ1901. Nezhivleva.Lab3.Task4.a.ixx

export module BPZ1901.Nezhivleva.Lab3.Task4:a;

import <cmath>;

namespace RBPO {

namespace Lab3 {

namespace Task4 {

export double a(int i);

}

}

}

Листинг 13 – файл BPZ1901. Nezhivleva.Lab3.Task4.f1.cpp

module BPZ1901.Nezhivleva.Lab3.Task4:f1;

namespace RBPO {

namespace Lab3 {

namespace Task4 {

double f1(double x) {

return (pow(x, 2.0) + 2.0 \* x - 3 + (x + 1) \* sqrt(pow(x, 2) - 9)) / (pow(x, 2.0) - 2.0 \* x - 3 + (x + 1) \* sqrt(pow(x, 2) - 9));

}

}

}

}

Листинг 14 – файл BPZ1901. Nezhivleva.Lab3.Task4.f1.ixx

export module BPZ1901.Nezhivleva.Lab3.Task4:f1;

import <cmath>;

namespace RBPO {

namespace Lab3 {

namespace Task4 {

export double f1(double x);

}

}

}

Листинг 15 – файл BPZ1901. Nezhivleva.Lab3.Task4.f2.cpp

module BPZ1901.Nezhivleva.Lab3.Task4:f2;

namespace RBPO {

namespace Lab3 {

namespace Task4 {

double f2(double x) {

if (x <= 4.0) return (sqrt(pow(x, 2.0) + 4.0));

else return cos(abs(2.0 + x));

}

}

}

}

Листинг 16 – файл BPZ1901. Nezhivleva.Lab3.Task4.f2.ixx

export module BPZ1901.Nezhivleva.Lab3.Task4:f2;

import <cmath>;

namespace RBPO {

namespace Lab3 {

namespace Task4 {

export double f2(double x);

}

}

}

Листинг 17 – файл BPZ1901. Nezhivleva.Lab3.Task4.f3.cpp

module BPZ1901.Nezhivleva.Lab3.Task4:f3;

namespace RBPO {

namespace Lab3 {

namespace Task4 {

double f3(int n) {

double sum = 0.0;

int i = 0;

do {

sum += a(i);

i++;

} while (i <= n);

return sum;

}

}

}

}

Листинг 18 – файл BPZ1901. Nezhivleva.Lab3.Task4.f3.ixx

export module BPZ1901.Nezhivleva.Lab3.Task4:f3;

import <cmath>;

import :a;

namespace RBPO {

namespace Lab3 {

namespace Task4 {

export double f3(int n);

}

}

}

Листинг 19 – файл BPZ1901. Nezhivleva.Lab3.Task4.f4.cpp

module BPZ1901.Nezhivleva.Lab3.Task4:f4;

namespace RBPO {

namespace Lab3 {

namespace Task4 {

double f4(double eps) {

double pred = a(0);

double sled = a(1);

double sum = pred + sled;

int i = 2;

do {

sum += a(i);

i++;

} while (abs(a(i) - a(i - 1)) > eps);

return sum;

}

}

}

}

Листинг 20– файл BPZ1901. Nezhivleva.Lab3.Task4.f4.ixx

export module BPZ1901.Nezhivleva.Lab3.Task4:f4;

import <cmath>;

import :a;

namespace RBPO {

namespace Lab3 {

namespace Task4 {

export double f4(double eps);

}

}

}

Листинг 21– файл BPZ1901. Nezhivleva.Lab3.Task4.ixx

export module BPZ1901.Nezhivleva.Lab3.Task4;

import <cmath>;

export import :f1;

export import :f2;

export import :a;

export import :f3;

export import :f4;

Задание 5

Объявления и определения функций , , , , поместить в одну единицу трансляции, описывающую интерфейс модуля (Module Interface Unit – файл с расширением \*.ixx).

При этом определение (реализацию) функций поместить в приватный фрагмент модуля (Module Private Fragment).

Экспортировать функции , , , . Само пространство имен НЕ экспортировать.

При необходимости подключить заголовочные файлы, используйте фрагмент глобального модуля (Global Module Fragment).

Листинг 22 – файл BPZ1901. Nezhivleva.Lab3.Task5.ixx

export module BPZ1901.Nezhivleva.Lab3.Task5;

import <cmath>;

export namespace RBPO {

namespace Lab3 {

namespace Task5 {

double f1(double x);

double f2(double x);

double a(int i);

double f3(int n);

double f4(double eps);

}

}

}

module:private;

double RBPO::Lab3::Task5::f1(double x) {

return (pow(x, 2.0) + 2.0 \* x - 3 + (x + 1) \* sqrt(pow(x, 2) - 9)) / (pow(x, 2.0) - 2.0 \* x - 3 + (x + 1) \* sqrt(pow(x, 2) - 9));

}

double RBPO::Lab3::Task5::f2(double x) {

if (x <= 4.0) return (sqrt(pow(x, 2.0) + 4.0));

else return cos(abs(2.0 + x));

}

double RBPO::Lab3::Task5::a(int i) {

return ((2 \* (i + 1)) / (2 + (i + 1) \* (1 - 1))) \* ((i % 2) ? -1.0 : 1.0);

}

double RBPO::Lab3::Task5::f3(int n) {

double sum = 0.0;

for (int i = 0; i <= n; i++) {

sum += a(i);

}

return sum;

}

double RBPO::Lab3::Task5::f4(double eps) {

double prev = a(0);

double next = a(1);

double sum = prev + next;

for (int i = 2; abs(a(i) - a(i - 1)) > eps; i++) {

sum += a(i);

}

return sum;

}

Задание 6

Разработать функцию main, демонстрирующую работу всех разработанных функций в заданиях 1-5.

Функция должна в цикле показывать меню и давать возможность выбрать задание и продемонстрировать работу всех функций из этого задания.

Листинг 23 – файл Main.cpp

#include <iostream>

import BPZ1901.Nezhivleva.Lab3.Task1;

import BPZ1901.Nezhivleva.Lab3.Task2;

import BPZ1901.Nezhivleva.Lab3.Task3;

import BPZ1901.Nezhivleva.Lab3.Task4;

import BPZ1901.Nezhivleva.Lab3.Task5;

using namespace std;

void task1(double x, int n, double eps);

void task2(double x, int n, double eps);

void task3(double x, int n, double eps);

void task4(double x, int n, double eps);

void task5(double x, int n, double eps);

int main() {

setlocale(LC\_ALL, "Russian");

double x;

double eps;

int n;

cout << "Здравствуйте!Введите данные для задачи:\n";

cout << "x = ";

cin >> x;

cout << "eps = ";

cin >> eps;

cout << "n = ";

cin >> n;

while (true) {

int choose = 1;

printf("Каким способом решим задачу?\n\tспособ №1 - Task1\n\tспособ №2- Task2\n\tспособ №3 - Task3\n\tспособ №4 - Task4\n\tспособ №5 - Task5\n\tВведите 6 для выхода из системы\nВыбранный способ: ");

cin >> choose;

printf("\n");

switch (choose) {

case 1:

task1(x, n, eps);

printf("\n");

break;

case 2:

task2(x, n, eps);

printf("\n");

break;

case 3:

task3(x, n, eps);

printf("\n");

break;

case 4:

task4(x, n, eps);

printf("\n");

break;

case 5:

task5(x, n, eps);

printf("\n");

break;

case 6:

printf("Вы завершили работу программы");

return 0;

default:

printf("Ошибка!\n");

}

}

return 0;

}

void task1(double x, int n, double eps) {

cout << "f1(" << x << ") : " << RBPO::Lab3::Task1::f1(x) << endl;

cout << "f2(" << x << ") : " << RBPO::Lab3::Task1::f2(x) << endl;

cout << "f3(" << n << ") : " << RBPO::Lab3::Task1::f3(n) << endl;

cout << "f4(" << eps << ") : " << RBPO::Lab3::Task1::f4(eps) << endl;

}

void task2(double x, int n, double eps) {

cout << "f1(" << x << ") : " << RBPO::Lab3::Task2::f1(x) << endl;

cout << "f2(" << x << ") : " << RBPO::Lab3::Task2::f2(x) << endl;

cout << "f3(" << n << ") : " << RBPO::Lab3::Task2::f3(n) << endl;

cout << "f4(" << eps << ") : " << RBPO::Lab3::Task2::f4(eps) << endl;

}

void task3(double x, int n, double eps) {

cout << "f1(" << x << ") : " << RBPO::Lab3::Task3::f1(x) << endl;

cout << "f2(" << x << ") : " << RBPO::Lab3::Task3::f2(x) << endl;

cout << "f3(" << n << ") : " << RBPO::Lab3::Task3::f3(n) << endl;

cout << "f4(" << eps << ") : " << RBPO::Lab3::Task3::f4(eps) << endl;

}

void task4(double x, int n, double eps) {

cout << "f1(" << x << ") : " << RBPO::Lab3::Task4::f1(x) << endl;

cout << "f2(" << x << ") : " << RBPO::Lab3::Task4::f2(x) << endl;

cout << "f3(" << n << ") : " << RBPO::Lab3::Task4::f3(n) << endl;

cout << "f4(" << eps << ") : " << RBPO::Lab3::Task4::f4(eps) << endl;

}

void task5(double x, int n, double eps) {

cout << "f1(" << x << ") : " << RBPO::Lab3::Task5::f1(x) << endl;

cout << "f2(" << x << ") : " << RBPO::Lab3::Task5::f2(x) << endl;

cout << "f3(" << n << ") : " << RBPO::Lab3::Task5::f3(n) << endl;

cout << "f4(" << eps << ") : " << RBPO::Lab3::Task5::f4(eps) << endl;

}

**Результат работы программы**

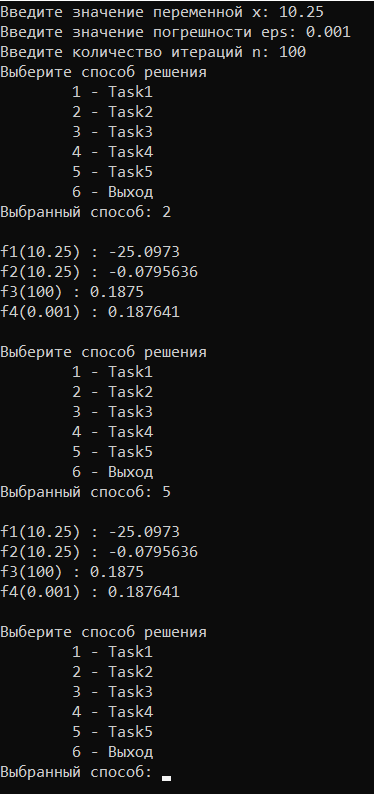


Рис.1-Результат работы программы