ФЕДЕРАЛЬНОЕ АГЕНСТВО ЖЕЛЕЗНОДОРОЖНОГО ТРАНСПОРТА

Федеральное государственное бюджетное образовательное учреждение высшего образования

«ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ПУТЕЙ СООБЩЕНИЯ Императора Александра I»

Кафедра «Информационные и вычислительные системы»

Дисциплина «Программирование С++»

**ОТЧЁТ**

**ПО ЛАБОРАТОРНОЙ РАБОТЕ № 1**

ВАРИАНТ 19

|  |  |
| --- | --- |
| Выполнил студент  Факультет: АИТ  Группа: ИВБ-211 | Шефнер А. |
| Проверил: | Проузин О.В. |

**Санкт-Петербург**

**2023**

Оценочный лист результатов ЛР № 1

Ф.И.О. студента \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Шефнер Альберт\_\_\_\_\_

Группа \_\_\_\_\_\_\_\_\_\_\_\_\_ИВБ-211\_\_\_\_\_\_\_\_\_\_\_\_

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **№**  **п/п** | **Материалы необходимые для оценки знаний, умений**  **и навыков** | **Показатель**  **оценивания** | **Критерии**  **Оценивания** | **Шкала оценивания** | **Оценка** |
| 1 | Лабораторная работа№ | Соответствие методике выполнения | Соответствует | 7 |  |
| Не соответствует | 0 |
| Срок выполнения | Выполнена в срок | 2 |  |
| Выполнена с опозданием на 2 недели | 0 |
| оформление | Соответствует требованиям | 1  0 |  |
| Не соответствует |  |
|  | **ИТОГО количество баллов** |  |  | 10 |  |

Доцент кафедры

«Информационные и вычислительные

системы» Проурзин О.В. «\_\_» \_\_\_\_\_\_\_\_\_\_2023 г.

**Цели работы:**

* освоить создание и вызов авторских функций;
* форматированный ввод-вывод в потоках;
* использование стандартных математических функций.

**Задание**

Написать программу, реализующую следующие пункты

1. Вычисление функции:

При этом, функция 

Выбор следующих блоков программы:

2.1. ввод x, вычисление и вывод значения f(x);

2.2. ввод x1, x2, вычисление и сравнение значений y(x1) и y(x2), вывод наибольшего

значения;

2.3. ввод x1, x2, d, вычисление и вывод значений y(x1), y(x1 + d), y(x1 + 2d), … ,

y(x2).

2.4. выход.

**Дополнительно**

* для выполнения пункта 2.3 использовать цикл …
* форматированный вывод по следующему правилу: …

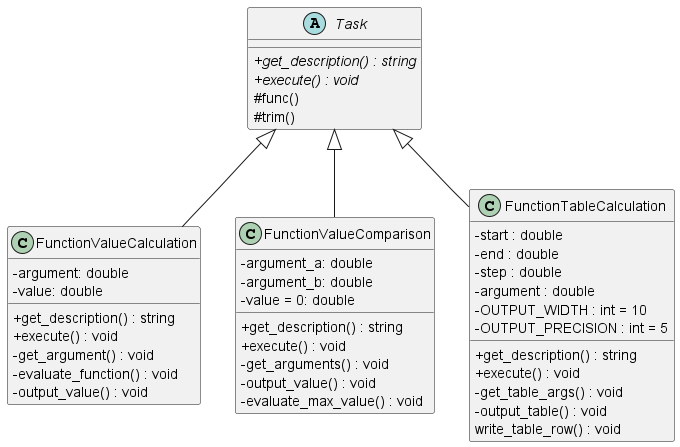
**Используемые средства**

В качестве интегрированной среды разработки использовалась JetBrains CLion.

Для работы в консоли с потоками ввода-вывода использовалась стандартная библиотека <iostream>, а также библиотека <iomanip> инструментов для работы с форматированием.

Был подключен заголовочный файл <math.h> для выполнения простых математических операций.

**UML-диаграмма программы:**

****

**Код программы с комментариями**

#include <iostream>

#include <iomanip>

#include <cmath>

#include <string>

#include <array>

#include "windows.h"

// я не знаю что мне тут комментировать

// наверное нет смысла писать,

// что func вычисляет значение функции

// или get\_description получает описание таска

// базовый класс Task, который содержит в себе 2 абстрактных метода для переопределения

// и 2 защищённых для вычисления функции

class Task {

public:

virtual std::string get\_description() = 0;

virtual void execute() = 0;

protected:

static double func(double argument) {

argument = trim(argument);

return (2 + argument \* argument) / (1 - argument) \* pow(cos(argument), 3);

}

static double trim(double argument) {

if (argument < -10) {

argument = -10;

} else if (argument > 10) {

argument = 10;

}

return argument;

}

};

// задание 1

class FunctionValueCalculation : public Task {

public:

std::string get\_description() override {

return "Вычислить значение функции в одной точке.";

}

void execute() override {

get\_argument();

evaluate\_function();

output\_value();

}

private:

double argument = 0;

double value = 0;

// по-хорошему бы в конструктор передать указатель на эту функцию, чтобы можно

// было задать другой вариант ввода значения, но во-первых мне лень,

// во-вторых это не нужно.

void get\_argument() {

std::cout << "Введите аргумент функции: ";

std::cin >> argument;

}

// evaluate переводится как вычислить

void evaluate\_function() {

value = func(argument);

}

// возводим объектно-ориентированный подход в абсолют там, где

// это совсем не нужно.

// Нам же надо из задачи на 30 строк сделать код на 300 строк, правильно?

void output\_value() const {

std::cout << "Значение функции: " << value << '\n';

}

};

// задание 2

class FunctionValueComparison : public Task {

public:

std::string get\_description() override {

return "Вычислить значение функции в двух точках и найти максимальное значение.";

}

void execute() override {

get\_arguments();

evaluate\_max\_value();

output\_value();

}

private:

double argument\_a = 0;

double argument\_b = 0;

double value = 0;

void get\_arguments() {

std::cout << "Введите два аргумента: ";

std::cin >> argument\_a >> argument\_b;

}

void output\_value() const {

std::cout << "Наибольшее значение функции: " << value << '\n';

}

void evaluate\_max\_value() {

value = std::max(func(argument\_a), func(argument\_b));

}

};

// задание 3

class FunctionTableCalculation : public Task {

public:

std::string get\_description() override {

return "Вычислить таблицу значений функции.";

}

void execute() override {

get\_table\_args();

output\_table();

}

private:

double start = 0;

double end = 0;

double step = 0;

double argument = 0;

const int OUTPUT\_WIDTH = 10;

const int OUTPUT\_PRECISION = 5;

void get\_table\_args() {

std::cout << "Введите начальное значение аргумента: ";

std::cin >> start;

std::cout << "Введите конечное значение аргумента: ";

std::cin >> end;

std::cout << "Введите шаг аргумента: ";

std::cin >> step;

}

// вывести таблицу

void output\_table() {

argument = start;

while (argument <= end) {

write\_table\_row();

argument += step;

}

}

// вывести одну строку таблицы ыы

void write\_table\_row() const {

std::cout << "Значение аргумента:" << std::setw(OUTPUT\_WIDTH) << std::setprecision(OUTPUT\_PRECISION) << argument;

std::cout << ", значение функции:" << std::setw(OUTPUT\_WIDTH) << std::setprecision(OUTPUT\_PRECISION)

<< func(argument) << '\n';

}

};

// объявление функций

void enable\_russian\_text();

void clear();

void display\_menu();

void get\_task\_number();

void execute\_task();

void pause();

// массив с тасками, к которому будет применён полиморфизм

std::array<Task \*, 3> tasks{new FunctionValueCalculation(), new FunctionValueComparison(),

new FunctionTableCalculation(),};

bool is\_running = true;

int task\_number;

// в данном случае он равен 3 так как в массиве 3 элемента (3 задания в ЛР)

const int EXIT\_NUMBER = tasks.size();

// int main() - функция main с типом возвращаемого значение int

int main() {

enable\_russian\_text();

while (is\_running) {

clear();

display\_menu();

get\_task\_number();

execute\_task();

pause();

}

return 0;

}

// реализация функций

void enable\_russian\_text() {

SetConsoleCP(CP\_UTF8);

SetConsoleOutputCP(CP\_UTF8);

}

// это выглядит довольно странно, но во-первых не всегда очевидно что

// такое system("cls"), а во-вторых не хочется мешать уровни абстракции

// внутри одной функции.

void clear() {

system("cls");

}

void display\_menu() {

std::cout << "Введите номер действия, которое хотите выполнить:\n";

for (int i = 0; i < tasks.size(); i++) {

std::cout << i + 1 << " - " << tasks[i]->get\_description() << "\n";

}

std::cout << EXIT\_NUMBER + 1 << " - Выход из программы\n";

}

void get\_task\_number() {

std::cin >> task\_number;

task\_number--;

}

void execute\_task() {

if (task\_number == EXIT\_NUMBER) {

is\_running = false;

return;

}

if (task\_number < 0 || task\_number >= tasks.size()) {

std::cout << "Неверный номер задание. Попробуйте ещё раз." << std::endl;

return;

}

tasks[task\_number]->execute();

}

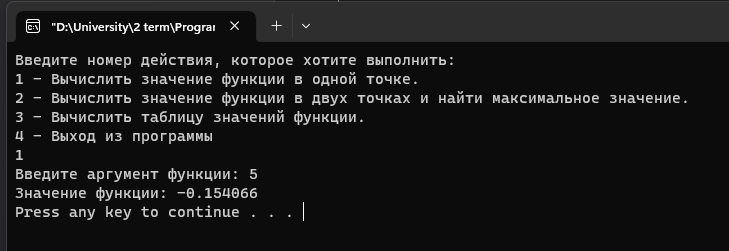
void pause() {

system("pause");

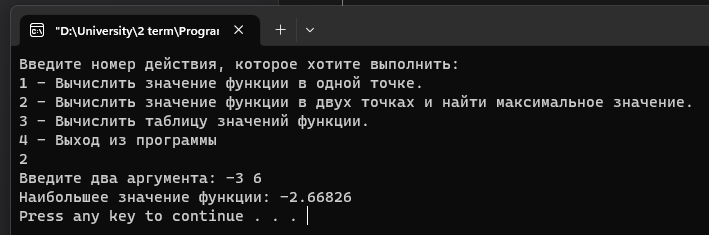
}

**Тестовые примеры**

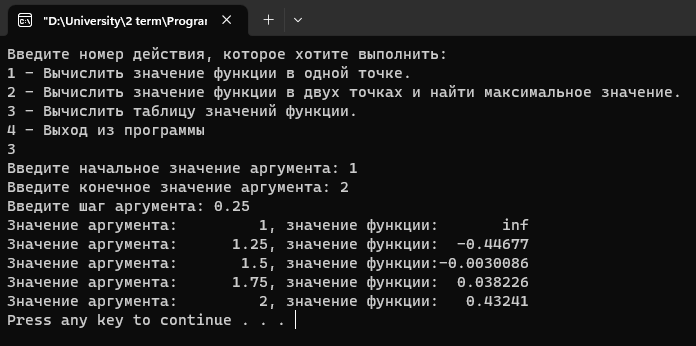
**2.1:**

****

**2.2:**

****

**2.3:**

****