

Министерство образования и науки Российской Федерации  
ФГБОУ ВО «НИУ «МЭИ»



---

Кафедра «Безопасности и информационных технологий»

Курсовая работа по теме:

---

**“Сравнение процедурного и объектно-ориентированного подхода  
на примере разработки приложения для решения конкретной  
задачи.”**

Решение двух уравнений

Выполнил:

Студент группы ИЭ-64-20

Бударин П. И. \_\_\_\_\_

"\_\_" \_\_\_\_\_ 2021 г.

Проверил:

Щёголев П. \_\_\_\_\_

"\_\_" \_\_\_\_\_ 2021 г.

Москва 2021 г.

## Оглавление

1 Введение.....	3
2 Цели и задачи курсовой работы .....	4
3 Условие задачи.....	5
4 Метод решения поставленной задачи.....	5
5 Описание данных (таблица).....	6
6 Блок схема основного вычислительного алгоритма.....	7
7 Процедурный подход: Описание подпрограмм, код консольного приложения с подпрограммами.....	8
8 Объектно-ориентированного подход: описание классов и методов, в том числе UML-диаграмма, иллюстрирующая иерархию разрабатываемых классов, код консольного приложения с классами .....	11
9 Код консольного приложения с классами:.....	12
10 Тесты .....	17
11 Вывод: Сравнение между процедурным и объектно-ориентированным подходами к программированию .....	20
12 Источники.....	21

## 1 Введение

Технология программирования – совокупность методов и инструментальных средств, используемых в процессе разработки программного обеспечения.

Структурное программирование подразумевает:

- точно обозначенные управляющие (базовые) структуры алгоритмов;
- соответствующее логике программы разбиение ее на программные блоки;
- автономные подпрограммы, в которых преимущественно используются локальные переменные;
- отсутствие (или, по крайней мере, ограниченное использование) операторов безусловного перехода – goto, break и др.

Объектно-ориентированное программирование (ООП)

- включает в себя лучшие идеи структурного программирования и дополняет их новыми мощными концепциями.
- ООП, так же как и структурное программирование, позволяет разложить задачу на подзадачи, но при этом каждая подзадача становится самостоятельным объектом, содержащим свои коды и данные.

## **2 Цели и задачи курсовой работы**

Целью является изучение объектно-ориентированного анализа и программирования на примере языка C++ и сравнение объектно-ориентированного подхода к разработке программ с структурным подходом.

Задачи:

1. Разработать консольное приложение с использованием функций и функций как параметров
2. Разработка консольного приложения с использованием принципов ООП. Реализовать наследование и виртуальный метод
3. Сравнить и сделать вывод из опыта использования этих двух подходов

### 3 Условие задачи

Методом дихотомии найдите корень уравнения  $F(x) = 0$  на отрезке  $[a, b]$  с заданной погрешностью  $E$ . Используйте в качестве  $F(x)$  формулу из Таблицы 1 и значения границ отрезка  $a = 0,1$  и  $b = 1$ . Для контроля правильности набора выражения: сумма значений выражения при  $x = 0.1, 0.2, 0.3, 0.4, 0.5$  приведена в третьем столбце.

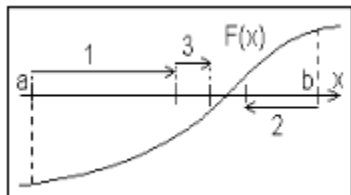
**Таблица 1**

Название

№	Формула $F(x)$	Сумма
1	$\frac{1,5}{\pi x} - 1 / \left  e^{0,8} - \sin \frac{1+x}{x} + \ln \sqrt[3]{x} \right $	8,64
2	$e^{ \sin x } + \frac{0,68 + \sqrt[3]{x}}{x} + \frac{1}{\cos^2 x} - \frac{10}{\pi x}$	-31,54

### 4 Метод решения поставленной задачи

При решении уравнения методом дихотомии на каждой итерации изменяются границы отрезка, внутри которого находится корень. Исходные границы задают так, чтобы знак функции  $F$  на границах был различен. Проверяют знак функции в средней точке  $x'$  текущего отрезка: точкой  $x'$  заменяют ту границу, где знак функции такой же. Подобное сокращение вдвое области выполняют многократно, пока ее размер не станет меньше  $E$  – допустимой погрешности значения корня. Функция  $F$  должна быть непрерывной в области поиска корня.



На рисунке показаны первые шаги рассмотренного процесса. На первом шаге заменяется левая граница. Второй шаг приведет к замене правой границы.

## 5 Описание данных (таблица)

Название переменной	Тип данных	Описание
a	double	Левая граница отрезка
b	double	Правая граница отрезка
name1	string	Имя функции
name2	string	Имя функции

## 6 Блок схема основного вычислительного алгоритма

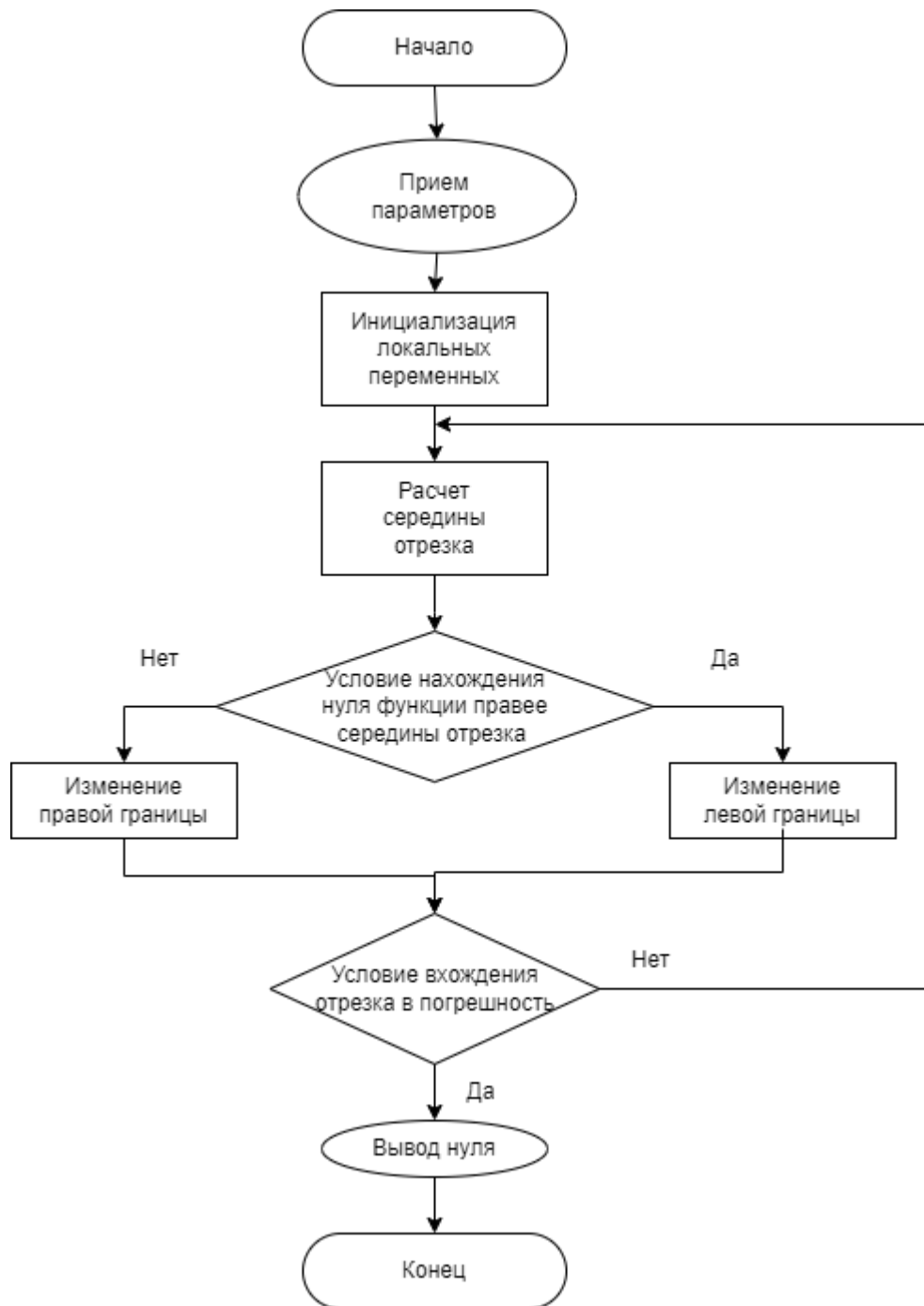


Рис. № 1 Блок схема основного вычислительного алгоритма

## 7 Процедурный подход: Описание подпрограмм, код консольного приложения с подпрограммами

Описание подпрограмм:

- Функция `f1`, принимающая и возвращающая тип `double`, рассчитывает значение первой функции в точке, переданной как параметр.
- Функция `f2`, принимающая и возвращающая тип `double`, рассчитывает значение второй функции в точке, переданной как параметр.
- Функция `findFIntervalNull`, принимающая указатель на заданную функцию и числа `a`, `b` типа `double` – границы отрезка, на котором происходит поиск нуля заданной функции, высчитывает и возвращает `x`, который является нулем заданной функции с погрешностью `E`.
- Функция `getFTestReport`, принимающая указатель на функцию, число `check_sum` типа `double` – сумму значений заданной функции при `x = [0.1, 0.2, 0.3, 0.4, 0.5]`, для проверки коректности заданной в программе функции и строку `name` – имя проверяемой функции, возвращает строку с отчетом о правильности заданной функции.

Код консольного приложения с подпрограммами:

```
#include <iostream>
#define _USE_MATH_DEFINES
#include <math.h>
#include <string>
using namespace std;

const double E = 0.0000001; //setting up the tolerance

double f1(double x);
double f2(double x);
double findFIntervalNull(double (*f)(double), double a, double b);
string getFTestReport(double (*f)(double), double check_sum, string name);

int main()
```



```

{
    double a = 0.1, b = 1; //setting the interval
    string name1 = "Function 1", name2 = "Function 2"; //set functions names

    //testing functions on input correctness using the sum of x in [0.1, 0.2, 0.3, 0.4,
0.5]
    cout << getFTestReport(*f1, 8.64, name1) << endl;
    cout << getFTestReport(*f2, -31.54, name2) << endl;

    //common header for print
    cout << "This is a root(s) of equation (function null on the interval[" << a << ",
" << b << "]),";
    cout << "calculated with a tolerance E = " << E << "\n";

    //print a calculated root of Function 1 on the interval[a, b]
    cout << name1 + ": \tx = " << findFIntervalNull(*f1, a, b) << endl;

    //print a calculated root of Function 2 on the interval[a, b]
    cout << name2 + ": \tx = " << findFIntervalNull(*f2, a, b) << endl;
}

double f1(double x) {
    return 1.5 / (M_PI * x) - 1 / abs(exp(0.8) - sin((1 + x) / x) + log(pow(x, 1.0 /
3))); //setting function 1, which null is calculated
}

double f2(double x) {
    return exp(abs(sin(x))) + (0.68 + pow(x, 1.0 / 3)) / x + 1 / pow(cos(x), 2) - 10 /
(M_PI * x); //setting function 2, which null is calculated
}

double findFIntervalNull(double (*f)(double), double a, double b) { //finding the
root of set function using bisection method
    double x;

    do {
        x = (b + a) / 2; //calculating the midpoint of the interval
        if ((*f)(x) * (*f)(a) > 0) { //if the function of midpoint and the left edge has
the same sign then
            a = x; //move left subinterval edge to the midpoint

```

```

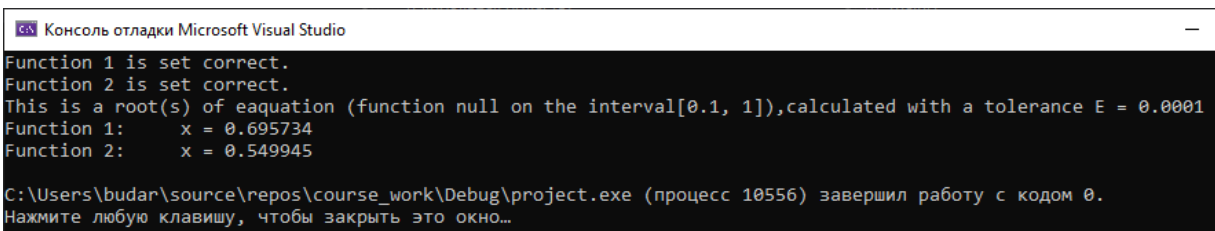
    }
    else { // or if the function has different signs then
        b = x; //move right subinterval edge to the midpoint
    }
} while ((b - a) > E); //while the tolerance E hasn't been reached

return x;
}

string getFTestReport(double (*f)(double), double check_sum, string name)
{
    double s = 0;
    for (double x = 0.1; x <= 0.5; x += 0.1) // calculate sum of several x to check the
set function
    {
        s += (*f)(x);
    }
    if (s == check_sum)
    {
        return name + " is set correct.";
    }
    else
    {
        return name + " is set incorrect.";
    }
}
}

```

Пример работы консольного приложения с использованием процедурного подхода к программированию. (Рис. №2)



```

Консоль отладки Microsoft Visual Studio
Function 1 is set correct.
Function 2 is set correct.
This is a root(s) of equation (function null on the interval[0.1, 1]),calculated with a tolerance E = 0.0001
Function 1:      x = 0.695734
Function 2:      x = 0.549945

C:\Users\budar\source\repos\course_work\Debug\project.exe (процесс 10556) завершил работу с кодом 0.
Нажмите любую клавишу, чтобы закрыть это окно...

```

Рис. № 2 Вывод информации функции

**8 Объектно-ориентированный подход: описание классов и методов, в том числе UML-диаграмма, иллюстрирующая иерархию разрабатываемых классов, код консольного приложения с классами**

UML-диаграмма, иллюстрирующая иерархию разрабатываемых классов (Рис. № 3)

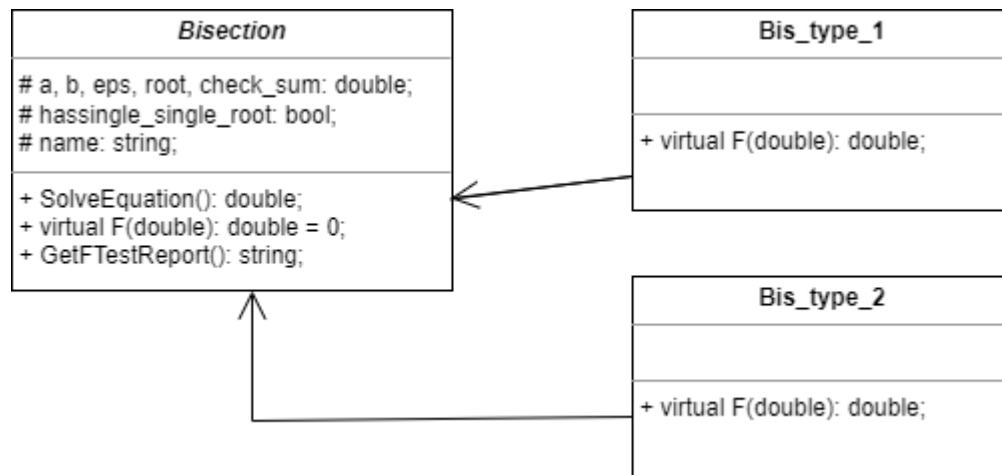


Рис. № 3 UML-диаграмма

Описание классов:

- Абстрактный класс **Bisection** предназначен для расчета значения нуля функции, на отрезке  $[a, b]$ , с заданной погрешностью `eps` и составления отчета о корректности задания функции в коде.
- Класс **Bis\_type\_1**, унаследованный от класса **Bisection**, переопределяет основной метод родительского класса задающий расчетную функцию.
- Класс **Bis\_type\_2**, унаследованный от класса **Bisection**, переопределяет основной метод родительского класса задающий расчетную функцию.

Описание методов:

- Метод `SolveEquation` класса **Bisection**, осуществляет расчеты нуля заданной функции на отрезке, с погрешностью `eps`, методом дихатомии.

- Виртуальный метод `F` класса `Bisection`, является абстрактным и переопределяется в классах наследниках, я целью задания расчетной функции.
- Метод `GetFTestReport` класса `Bisection`, возвращает строку о результате проверки корректности задания расчетной функции, по средствам сравнения суммы значений заданной функции при  $x = [0.1, 0.2, 0.3, 0.4, 0.5]$  с заданной контрольной суммой.

## 9 Код консольного приложения с классами:

```
#include <iostream>
#define _USE_MATH_DEFINES
#include <math.h>
#include <string>

using namespace std;

class Bisection {
protected:
    double a, b, eps, root = 0;
    bool has_single_root = true;
    string name;
    double check_sum;
public:
    Bisection(double a, double b, double eps, double check_sum, string name);
    //parent constructor
    double SolveEquation(); //method solving the equation  $F(x) = 0$ 
    virtual double F(double x) = 0; //parent class doesn't have a realisation of this
    method, so it's = 0. Abstract method
    string GetFTestReport();
};

class Bis_type_1: public Bisection
{
public:
    Bis_type_1(double a, double b, double eps, double check_sum, string name)
    :Bisection(a, b, eps, check_sum, name) {}; //child constructor
```

```

        virtual double F(double x); //overloading parent method that set the type 1 of
an equation
    };

class Bis_type_2: public Bisection
{
public:
    Bis_type_2(double a, double b, double eps, double check_sum, string name)
:Bisection(a, b, eps, check_sum, name) {}; //child constructor
    virtual double F(double x); //overloading parent method that set the type 2 of
an equation
};

int main()
{
    double a = 0.1, b = 1; //setting the interval
    double eps = 0.00001; //set tolerance
    string name1 = "Function 1", name2 = "Function 2"; //set functions names
    Bis_type_1* eq1 = new Bis_type_1(a, b, eps, 8.64, name1);
    Bis_type_2* eq2 = new Bis_type_2(a, b, eps, 31.54, name2);

    //common header for print
    cout << "This is a root(s) of equation (function null on the interval[" << a
<< ", " << b << "]),";
    cout << "calculated with a tolerance E = " << eps << "\n";

    //testing functions on input correctness using the sum of x in [0.1, 0.2, 0.3,
0.4, 0.5]
    cout << eq1->GetFTestReport() << endl;
    cout << eq2->GetFTestReport() << endl;

    //print a calculated root of Function 1 on the interval[a, b]
    cout << name1 + ": \tx = " << eq1->SolveEquation() << endl;

    //print a calculated root of Function 2 on the interval[a, b]
    cout << name2 + ": \tx = " << eq2->SolveEquation() << endl;
}

```

```

Bisection::Bisection(double a, double b, double eps, double check_sum, string
name) //parent constructor description
{
    this->a = a;
    this->b = b;
    this->eps = eps;
    this->check_sum = check_sum;
    this->name = name;
}
string Bisection::GetFTestReport()
{
    double s = 0;
    for (double x = 0.1; x <= 0.5; x += 0.1) // calculate sum of several x to check
the set function
    {
        s += F(x);
    }
    if (s == check_sum)
    {
        return name + " is set correct.";
    }
    else
    {
        return name + " is set incorrect.";
    }
}

double Bisection::SolveEquation() //bisection method description
{
    double midpoint = (a + b) / 2; //calculating the midpoint of the interval
    if (F(a) * F(b) > 0) // if an equation has even amount of roots including none,
then
    {
        has_single_root = false;
        return a - 1; //return impossible answer
    }
    else
    {
        has_single_root = true;
        while (b - a > eps) //while the tolerance E hasn't been reached

```

```

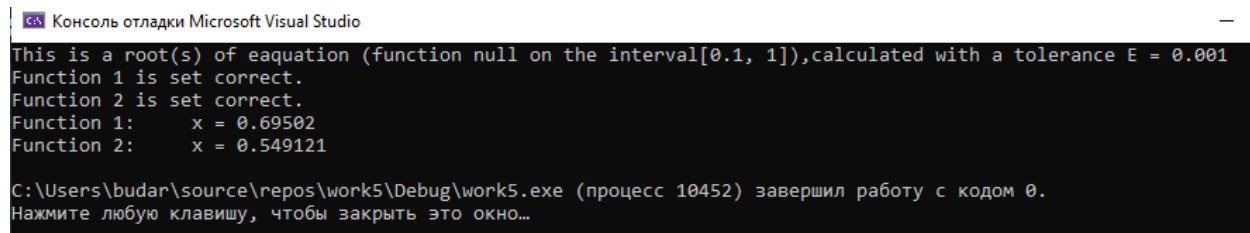
        {
            midpoint = (a + b) / 2; //calculating the midpoint of the interval
            if (F(a) * F(midpoint) > 0) //if the function of midpoint and the
left edge has the same sign then
                {
                    a = midpoint; //move left subinterval edge to the
midpoint
                }
            else // or if the function has different signs then
            {
                b = midpoint; //move right subinterval edge to the
midpoint
            }
        }
    }
    return midpoint;
}

double Bis_type_1::F(double x) //setting child 1 function, which null is calculated
{
    return 1.5 / (M_PI * x) - 1 / abs(exp(0.8) - sin((1 + x) / x) + log(pow(x, 1.0 /
3))); //equation 1
}

double Bis_type_2::F(double x) //setting child 2 function, which null is calculated
{
    return exp(abs(sin(x))) + (0.68 + pow(x, 1.0 / 3)) / x + 1 / pow(cos(x), 2) -
10 / (M_PI * x); //equation 2
}

```

Пример работы консольного приложения с использованием объектно-ориентированного подхода к программированию. (Рис. №4)



```
Консоль отладки Microsoft Visual Studio

This is a root(s) of equation (function null on the interval[0.1, 1]),calculated with a tolerance E = 0.001
Function 1 is set correct.
Function 2 is set correct.
Function 1:      x = 0.69502
Function 2:      x = 0.549121

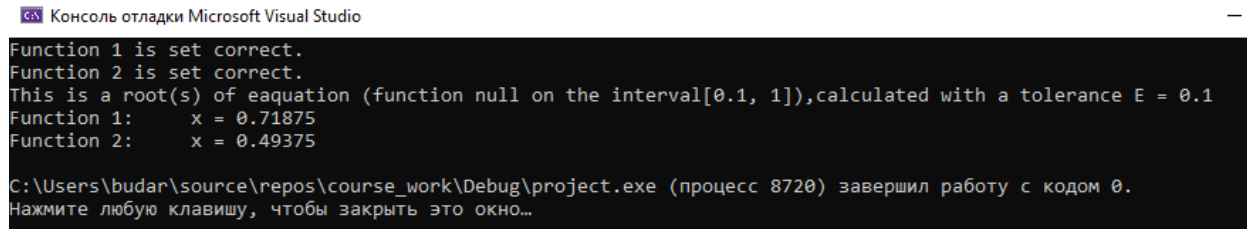
C:\Users\budar\source\repos\work5\Debug\work5.exe (процесс 10452) завершил работу с кодом 0.
Нажмите любую клавишу, чтобы закрыть это окно...
```

Рис. № 4 Вывод информации



## 10 Тесты

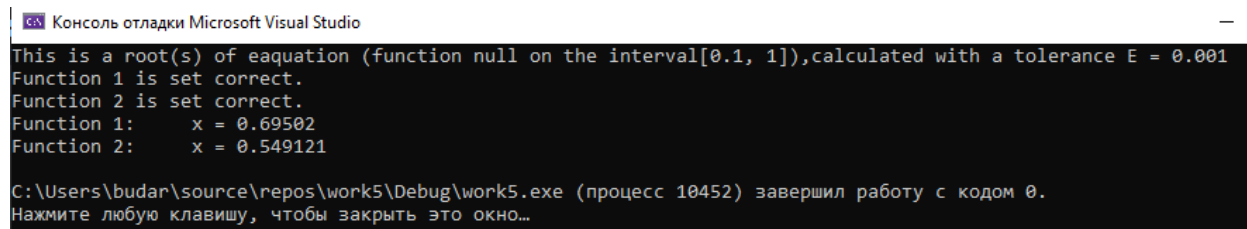
Результаты тестирования консольного приложения с использованием процедурного подхода к программированию, при различных значениях  $E$  - погрешности. (Рис. №5, Рис. №6, Рис. №7)



```
Консоль отладки Microsoft Visual Studio
Function 1 is set correct.
Function 2 is set correct.
This is a root(s) of equation (function null on the interval[0.1, 1]),calculated with a tolerance E = 0.1
Function 1:    x = 0.71875
Function 2:    x = 0.49375

C:\Users\budar\source\repos\course_work\Debug\project.exe (процесс 8720) завершил работу с кодом 0.
Нажмите любую клавишу, чтобы закрыть это окно...
```

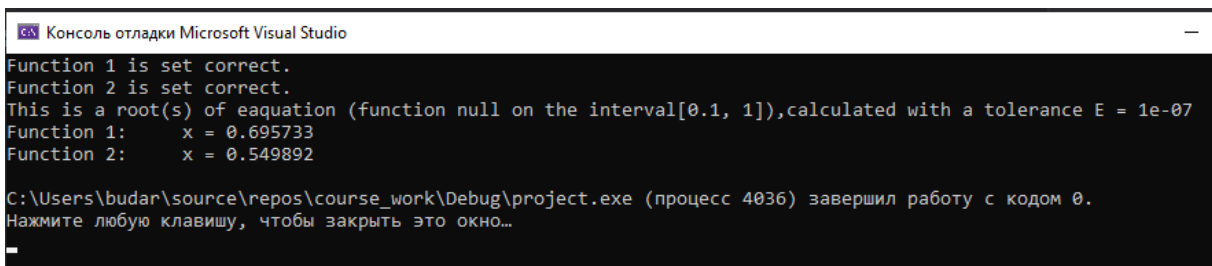
Рис. № 5 Результат теста 1



```
Консоль отладки Microsoft Visual Studio
This is a root(s) of equation (function null on the interval[0.1, 1]),calculated with a tolerance E = 0.001
Function 1 is set correct.
Function 2 is set correct.
Function 1:    x = 0.69502
Function 2:    x = 0.549121

C:\Users\budar\source\repos\work5\Debug\work5.exe (процесс 10452) завершил работу с кодом 0.
Нажмите любую клавишу, чтобы закрыть это окно...
```

Рис. № 6 Результат теста 2



```
Консоль отладки Microsoft Visual Studio
Function 1 is set correct.
Function 2 is set correct.
This is a root(s) of equation (function null on the interval[0.1, 1]),calculated with a tolerance E = 1e-07
Function 1:    x = 0.695733
Function 2:    x = 0.549892

C:\Users\budar\source\repos\course_work\Debug\project.exe (процесс 4036) завершил работу с кодом 0.
Нажмите любую клавишу, чтобы закрыть это окно...
```

Рис. № 7 Результат теста 3

Результаты тестирования консольного приложения с использованием объектно-ориентированного подхода к программированию, при различных значениях  $\epsilon$  - погрешности. (Рис. №8, Рис. №9, Рис. №10)

```
Консоль отладки Microsoft Visual Studio
This is a root(s) of equation (function null on the interval[0.1, 1]),calculated with a tolerance E = 0.01
Function 1 is set correct.
Function 2 is set correct.
Function 1:      x = 0.697656
Function 2:      x = 0.542969

C:\Users\budar\source\repos\work5\Debug\work5.exe (процесс 13404) завершил работу с кодом 0.
Нажмите любую клавишу, чтобы закрыть это окно...
```

Рис. № 8 Результат теста 1

```
Консоль отладки Microsoft Visual Studio
This is a root(s) of equation (function null on the interval[0.1, 1]),calculated with a tolerance E = 0.001
Function 1 is set correct.
Function 2 is set correct.
Function 1:      x = 0.69502
Function 2:      x = 0.549121

C:\Users\budar\source\repos\work5\Debug\work5.exe (процесс 10452) завершил работу с кодом 0.
Нажмите любую клавишу, чтобы закрыть это окно...
```

Рис. № 9 Результат теста 2


```
Консоль отладки Microsoft Visual Studio
This is a root(s) of equation (function null on the interval[0.1, 1]),calculated with a tolerance E = 1e-05
Function 1 is set correct.
Function 2 is set correct.
Function 1:      x = 0.695727
Function 2:      x = 0.549897


C:\Users\budar\source\repos\work5\Debug\work5.exe (процесс 5092) завершил работу с кодом 0.
Нажмите любую клавишу, чтобы закрыть это окно...
```

Рис. № 10 Результат теста 3


Результаты расчетов нулей заданных функций на сайте WolframAlfa, для проверки результатов работы консольных приложений. (Рис. №11, Рис. №12)

1.5 / (pi \* x) - 1 / abs(exp(0.8) - sin((1 + x) / x) + log(pow(x, 1.0 / 3))) = 0; 0.1 < x < 1

 NATURAL LANGUAGE

 MATH INPUT

 EXTENDED KEYBOARD

 EXAMPLES



Assuming "log" is the natural logarithm | Use [the base 10 logarithm](#) instead

Input

$$\left\{ \frac{1.5}{\pi x} - \frac{1}{\left| \exp(0.8) - \sin\left(\frac{1+x}{x}\right) + \log\left(\sqrt[3]{x}\right) \right|} = 0, 0.1 < x < 1 \right\}$$

log(x) is th

|z| is the

Result

$$\left\{ \frac{0.477465}{x} - \frac{1}{\left| \log(x^{0.333333}) - \sin\left(\frac{x+1}{x}\right) + 2.22554 \right|} = 0, 0.1 < x < 1 \right\}$$


Solution

$x \approx 0.695733$

Рис. № 11 Ноль функции 1

exp(abs(sin(x))) + (0.68 + pow(x, 1.0 / 3)) / x + 1 / pow(cos(x), 2) - 10 / (pi \* x) = 0; 0.1 < x < 1

 NATURAL LANGUAGE

 MATH INPUT

 EXTENDED KEYBOARD

 EXAMPLES

 UPL

Input

$$\left\{ \exp(|\sin(x)|) + \frac{0.68 + \sqrt[3]{x}}{x} + \frac{1}{\cos^2(x)} - \frac{10}{\pi x} = 0, 0.1 < x < 1 \right\}$$

|z| is the abs

Result

$$\left\{ e^{|\sin(x)|} + \frac{x^{0.333333} + 0.68}{x} - \frac{10}{\pi x} + \sec^2(x) = 0, 0.1 < x < 1 \right\}$$

sec(x) is the

Real solution

$x \approx 0.549892$

Рис. № 12 Ноль функции 2

## **11 Вывод: Сравнение между процедурным и объектно-ориентированным подходами к программированию**

Объектно-ориентированный подход более естественен, чем функциональный, поскольку в первую очередь ориентирована на объектное восприятие мира, что проще понять человеческому восприятию.

В нашей задаче мы не смогли ощутить преимущество подхода ООП по сокращению кода, так как задача была не достаточно большой. Но удалось сократить количество передаваемых параметров, как при вызове, так и при описании функций, что ведет к снижению количества ошибок и упрощению отлаживания кода.

## 12 Источники

Настоящий курсовой проект создан на основе следующего набора информационных материалов:

Приложения и среды разработки:

- Microsoft Visual Studio 2022 Community – среда разработки на языке C++
- Сайты: draw.io - Онлайн конструктор диаграмм.

Литература:

- The C++ Programming Language 5th edition by PROCODE Publishing
- Б. Страуструп. Язык программирования C++. — Издательство: Бином, 2011 – 546 с. В.С. Зубов, В.С. Батасова.
- Сборник задач по базовой компьютерной подготовке: учебное пособие по курсу «Информатика». – М.: Издательский дом МЭИ, 2007 – 124 с