



МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ  
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ  
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ  
«ДОНСКОЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»  
(ДГТУ)

Факультет «Информатика и вычислительная техника»  
(наименование факультета)

Кафедра «Программное обеспечение вычислительной техники и автоматизированных систем»  
(наименование кафедры)

Зав. кафедрой

«ПОВТиАС»

В.В. Долгов

(подпись)

(И.О.Ф.)

«\_\_\_\_\_» 2025 г.

## ОТЧЕТ

по учебной технологической (проектно-технологической) практике  
вид практики

на базе научно-образовательного производственного центра «Визуализация, анализ и защита  
информации» (НОПЦ «ВАиЗИ»)

наименование базы практики

Обучающийся

\_\_\_\_\_

подпись, дата

В. А. Щегольков

И.О.Ф.

Обозначение отчета

УП.990000.000

Группа ВМО11

Направление 02.03.03 Математическое обеспечение и администрирование информационных  
систем

код

наименование направления подготовки

Профиль Большие данные и машинное обучение

Руководитель практики:

от предприятия руководитель

должность

подпись, дата

В.В. Долгов

имя, отчество, фамилия

от кафедры доцент

должность

подпись, дата

Т.А. Медведева

имя, отчество, фамилия

Оценка

дата

подпись преподавателя

Ростов-на-Дону  
2025 г.



МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ  
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ  
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ  
«ДОНСКОЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»  
(ДГТУ)

Факультет «Информатика и вычислительная техника»  
(наименование факультета)

Кафедра «Программное обеспечение вычислительной техники и автоматизированных систем»  
(наименование кафедры)

## ЗАДАНИЕ

на учебную технологическую (проектно-технологическую) практику  
вид практики

в научно-образовательном производственном центре «Визуализация, анализ и защита  
информации» (НОПЦ «ВАиЗИ»)

наименование базы практики

в период с «23» июня 2025 г. по «05» июля 2025 г.

Обучающийся Щегольков Владимир Анатольевич

Обозначение отчета УП.990000.000 Группа ВМО11

Срок представления отчета на кафедру «05» июля 2025 г.

Содержание индивидуального задания:

Программная реализация численного решения нелинейных уравнений методом половинного деления

Руководитель практики от  
кафедры

\_\_\_\_\_

подпись, дата

Т.А. Медведева  
И.О.Ф.

Задание принял к исполнению

\_\_\_\_\_

подпись, дата

В.А. Щегольков  
И.О.Ф.



МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ  
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ  
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ  
«ДОНСКОЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»  
(ДГТУ)

Факультет «Информатика и вычислительная техника»  
(наименование факультета)

Кафедра «Программное обеспечение вычислительной техники и автоматизированных систем»  
(наименование кафедры)

Зав. кафедрой

«ПОВТиАС»

В.В. Долгов

(подпись)

(И.О.Ф.)

«\_\_\_\_\_» 2025 г.

### Рабочий график (план) проведения практики

№	Мероприятие	Срок выполнения
1	Прохождение вводного и первичного инструктажа по охране труда на рабочем месте, и инструктажа по пожарной безопасности на объекте	23.06.2025
2	Получение индивидуального задания и постановка задачи	23.06.2025
3	Изучение теоретического материала по теме численного решения нелинейных уравнений	С 24.06.2025 – 26.06.2025
4	Алгоритмическое конструирование численного решения нелинейных уравнений методом половинного деления	С 27.06.2025 – 29.06.2025
5	Выбор среды разработки и реализация программного средства	С 30.06.2025 – 02.07.2025
6	Тестирование программного средства, составление отчета о проделанной работе	С 03.07.2025 – 04.07.2025
7	Защита итогового отчета по практике	05.07.2025

Руководитель практики:

от предприятия руководитель \_\_\_\_\_  
должность \_\_\_\_\_ подпись, дата \_\_\_\_\_  
имя, отчество, фамилия \_\_\_\_\_

от кафедры доцент \_\_\_\_\_  
должность \_\_\_\_\_ подпись, дата \_\_\_\_\_  
имя, отчество, фамилия \_\_\_\_\_

Ростов-на-Дону  
2025 г.

## ДНЕВНИК ПРОХОЖДЕНИЯ ПРАКТИЧЕСКОЙ ПОДГОТОВКИ

Дата	Место работы	Выполняемые работы	Оценка руководителя
23.06.2025	НОПЦ «ВАиЗИ»	Знакомство с предприятием, прохождение вводного инструктажа	
23.06.2025	НОПЦ «ВАиЗИ»	Ознакомление с территорией предприятия, прохождение первичного инструктажа по ТБ, ПБ	
23.06.2025	НОПЦ «ВАиЗИ»	Получение индивидуального задания и постановка задачи	
С 24.06.2025 по 26.06.2025	НОПЦ «ВАиЗИ»	Изучение теоретических материалов по теме численного решения нелинейных уравнений	
С 27.06.2025 по 29.06.2025	НОПЦ «ВАиЗИ»	Алгоритмическое конструирование численного решения нелинейных уравнений методом половинного деления	
С 30.06.2025 по 02.07.2025	НОПЦ «ВАиЗИ»	Выбор среды разработки и реализация программного средства	
С 03.07.2025 по 04.07.2025	НОПЦ «ВАиЗИ»	Проверка работоспособности программы, составление отчета о проделанной работе	
05.07.2025	НОПЦ «ВАиЗИ»	Защита итогового отчета по практике	

Руководитель практики:

от предприятия \_\_\_\_\_ руководитель \_\_\_\_\_  
должность \_\_\_\_\_ подпись, дата \_\_\_\_\_  
имя, отчество, фамилия \_\_\_\_\_

## ОТЗЫВ – ХАРАКТЕРИСТИКА

Обучающийся Щегольков В.А  
фамилия, имя, отчество

1 курса группы ВМО11 кафедра «Программное обеспечение вычислительной техники и автоматизированных систем»

Вид практики учебная технологическая (проектно-технологическая) практика

Наименование места практики научно-образовательный производственный центр «Визуализация, анализ и защита информации» (НОПЦ «ВАиЗИ»)  
наименование предприятия, структурного подразделения

Обучающийся выполнил задания программы практики

Изучена и освоена теоретическая информация о численных методах решения нелинейных уравнений. Реализовано и протестировано программное средство с удобным консольным интерфейсом для численного решения нелинейных уравнений методом половинного деления.

Дополнительно ознакомился/изучил

Численные методы решения нелинейных уравнений, метод половинного деления, библиотеки Matplotlib, NumPy, SymPy

Заслуживает оценки

---

Руководитель практики от  
предприятия

«  » 2025 г.

Б.П.

## Содержание

Введение.....	7
1 Теоретический обзор.....	8
1.1 Основные понятия нелинейных уравнений .....	8
1.2 Метод половинного деления. ....	9
1.3 Выводы по главе .....	11
1.4 Постановка задачи .....	11
2 Алгоритмическое конструирование .....	13
2.1 Схема реализации метода половинного деления.....	14
2.2 Выводы по главе .....	14
3 Программное конструирование .....	16
3.1 Выбор языка программирования и среды разработки .....	16
3.2 Описание основных классов и методов программного средства .....	18
3.3 Описание программы .....	19
3.4 Выводы по главе .....	21
4 Тестирование программного средства .....	22
4.1 Результат работы программы .....	22
4.2 Выводы по главе .....	24
Заключение .....	25
Перечень использованных информационных ресурсов.....	26
Приложение А Исходный код программного средства .....	27

Изм.	Лист	№ докум.	Подпись	Дата				
Разраб.	Щегольков В.А.							
Пров.	Медведева Т. А.							
Н. Контр.								
Утв.	Долгов В. В.							
<b>УП.990000.000</b>								
Программная реализация численного решения нелинейных уравнений методом половинного деления					Лит.	Лист	Листов	
						6	30	
					ДГТУ кафедра «ПОВТиАС»			

## **Введение**

При решении задач в таких областях, как физика, электротехника, экономика и вычислительная математика, часто возникает необходимость определять корни нелинейных уравнений. Эти уравнения занимают центральное место в математическом моделировании, так как описывают сложные взаимосвязи между параметрами систем, которые не удается выразить через линейные зависимости.

Нелинейные уравнения отличаются тем, что искомая переменная входит в функцию в нелинейной форме, например, в виде многочленов высокой степени, тригонометрических, экспоненциальных или логарифмических выражений. Такие уравнения, как правило, не поддаются точному аналитическому решению, что требует применения численных методов.

Существует ряд численных методов, позволяющих находить приближенные значения корней с заданной точностью. Эти методы незаменимы в ситуациях, когда аналитическое решение либо невозможно, либо слишком трудоемко. Они обеспечивают надежные алгоритмы для вычисления корней уравнений разной сложности и активно используются в инженерных, физических и экономических задачах. К численным методам относятся метод Ньютона (касательных), метод половинного деления, метод хорд и метод секущих. Каждый из них имеет свои особенности, область применения и скорость сходимости, что позволяет выбрать наиболее подходящий метод в зависимости от специфики задачи.

В данной работе рассматривается один из численных методов решения нелинейных уравнений — метод половинного деления.

Изм.	Лист	№ докум.	Подпись	Дата	Лист	УП.990000.000	7

# 1 Теоретический обзор

В данном разделе рассмотрена теоретическая информация о нелинейных уравнениях и численных методах их решения. Подробно описывается метод половинного деления.

## 1.1 Основные понятия нелинейных уравнений

Нелинейным уравнением называется уравнение, в котором искомая переменная или её функции входят в нелинейной форме, то есть не только в первой степени или не в виде простой суммы/разности. В общем случае нелинейное уравнение может быть представлено в виде:

$$f(x) = 0 \quad (1)$$

где  $f(x)$  - определена и непрерывна ( $f(x) \subset C(a, b)$ ) в некотором конечном или бесконечном интервале  $(a, b)$ .

Всякое число  $x^*$ , удовлетворяющее (1), называется его корнем. Корни (1) считаются изолированными, если в некоторой окрестности  $|x - x^*| < \varepsilon$  не существует других корней.

Нелинейные уравнения могут быть классифицированы по следующим признакам:

1. Алгебраические уравнения – включают только алгебраические функции (целые, рациональные или иррациональные).
2. Трансцендентные уравнения – содержат хотя бы одну трансцендентную функцию (например, тригонометрические, экспоненциальные, логарифмические).

Трансцендентные уравнения, включающие алгебраические, тригонометрические или экспоненциальные функции от неизвестной  $x$ , часто имеют бесконечное число корней. Необходимость их решения возникает, например, при анализе устойчивости систем, моделировании парожидкостного равновесия, изучении динамики популяций или расчёте задач, связанных с законами сохранения энергии.

Изм.	Лист	№ докум.	Подпись	Дата	Лист	8
					УП.990000.000	

Методы решений делятся на:

1. Прямые (аналитические), позволяющие выразить корни в виде точной формулы (например, для простых алгебраических, тригонометрических, логарифмических или экспоненциальных уравнений).
2. Итерационные (численные), позволяющие за конечное число шагов находить приближённое значение корня с заданной точностью.

Этапы приближённого метода можно разделить на два пункта:

1. Отделение корней – установление возможно тесного промежутка  $[\alpha, \beta] \ni x^* (f(x^*) = 0)$  на котором содержится один и только один корень  $x^*$  уравнения (1).
2. Уточнение корней – доведение корней до заданной степени точности численными методами решения.

К основным численным методам решения нелинейных уравнений относятся метод половинного деления, метод хорд, метод Ньютона, метод простых итераций и метод секущих.

## 1.2 Метод половинного деления

Метод половинного деления – один из наиболее надежных численных методов решения нелинейных уравнений. Он основан на идее последовательного сужения интервала, содержащего корень, путем деления его пополам.

Пусть корень  $x^*$  нелинейного уравнения  $f(x) = 0$  отделен на отрезке  $[a, b]$ , и требуется вычислить его с заданной точностью  $\varepsilon$ , то есть определить  $x^* \approx x_n$ , где  $|x_n - x^*| < \varepsilon$ . Предполагается, что функция  $f(x)$  непрерывна на  $[a, b]$ , а значения  $f(a)$  и  $f(b)$  имеют противоположные знаки:  $f(a) \cdot f(b) < 0$ . Тогда на отрезке  $[a, b]$ , существует корень. На каждом шаге отрезок делится

Изм.	Лист	№ докум.	Подпись	Дата
------	------	----------	---------	------

УП.990000.000

Лист

9

пополам, вычисляется средняя точка  $c = \frac{a+b}{2}$ , и проверяется знак функции  $f(c)$ :

$$x_n = \frac{a+b}{2}, n = 0, 1, 2, \dots$$

Геометрически метод половинного деления можно представить, выбрав отрезок  $[a, b]$ , где  $f(a) \cdot f(b) < 0$ , и разделив его пополам. За новое приближение корня  $x^*$  принимается середина  $c$ , если  $f(c) = 0$ , или выбирается новый отрезок  $[a, c]$  или  $[c, b]$ , где выполняется условие  $f(a) \cdot f(c) < 0$  или  $f(c) \cdot f(b) < 0$ . Далее процесс повторяется с новым отрезком. Условие окончания итераций:

$$|b - a| < 2\varepsilon \text{ или } |f(x_n)| < \varepsilon$$

Геометрическая иллюстрация метода половинного деления представлена на рисунке 1.

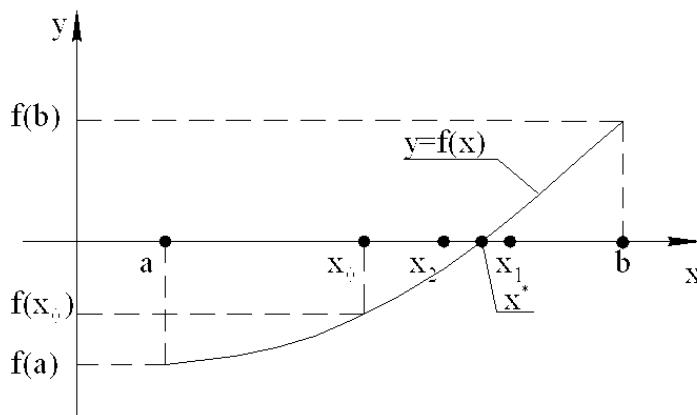


Рисунок 1 - Геометрическая иллюстрация метода половинного деления

На рисунке отрезок  $[a, b]$  делится пополам, и новый отрезок выбирается в зависимости от знака функций в средней точке  $c$ :

$$c = \frac{a + b}{2}$$

Если на отрезке  $[a, b]$  функция  $f(x)$  непрерывна и  $f(a) \cdot f(b) < 0$ , метод гарантирует сходимость к корню. Начальные точки  $a$  и  $b$  выбираются так, чтобы выполнялось условие противоположности знаков функции на концах отрезка:

$$f(a) \cdot f(b) < 0$$

Изм.	Лист	№ докум.	Подпись	Дата

УП.990000.000

На рисунке 1 график  $f(x)$  пересекает ось  $Ox$  в точке  $x^*$ . На каждом шаге длина отрезка уменьшается вдвое, что обеспечивает сходимость. Метод половинного деления отличается простотой и надежностью, так как не требует вычисления производных. Его недостатками являются относительно медленная сходимость, так как длина отрезка уменьшается линейно, отсутствие возможности нахождения корней четной кратности, не обобщается на систему уравнений.

### 1.3 Выводы по главе

В представленном разделе изложены основные понятия и классификация решения нелинейных уравнений с использованием численных методов. Основной акцент сделан на описании метода половинного деления, обладающего простотой реализации и надежностью. Данный алгоритм обеспечивает гарантированную сходимость при соблюдении ключевых условий: непрерывности функции на заданном отрезке и различия знаков функции на его границах. Однако следует отметить, что метод характеризуется линейной скоростью сходимости, что обуславливает его меньшую эффективность по сравнению с итерационными методами более высокого порядка, в частности, с методом Ньютона.

### 1.4 Постановка задачи

В результате теоретического обзора поставлена цель: разработать программное средство, позволяющее решать нелинейные уравнения численным методом половинного деления с визуализацией полученных результатов. Консольный интерфейс программы должен обеспечивать возможность ввода произвольного нелинейного уравнения и точности вычислений, задания границ интервала, автоматического выполнения итерационного процесса, а также построения графика исходной функции и

Изм.	Лист	№ докум.	Подпись	Дата	Лист 11
					УП.990000.000

отображения таблицы приближённых значений, полученных в ходе вычислений.

Для достижения поставленной цели необходимо решить следующие задачи:

- реализовать вычислительные алгоритмы численного решения;
- определить среду разработки и язык программирования;
- реализовать функциональные модули: численных расчётов, графического отображения результатов, табличного представления данных;
- провести тестирование ПС на нескольких контрольных примерах;
- в соответствии с отчетом по тестированию выполнить корректировку ПС;
- разработать сопроводительную документацию: отчет.

Изм.	Лист	№ докум.	Подпись	Дата

**УП.990000.000**

Лист

12

## 2 Алгоритмическое конструирование

В данном разделе исследуется алгоритмическая реализация метода половинного деления - численного подхода к решению нелинейных уравнений. Представленная методика позволяет разработать программное средство для приближенного нахождения корней нелинейных уравнений посредством итерационного процесса.

### 2.1 Схема реализации метода половинного деления

На рисунке 2 продемонстрирована схема реализации метода половинного деления.

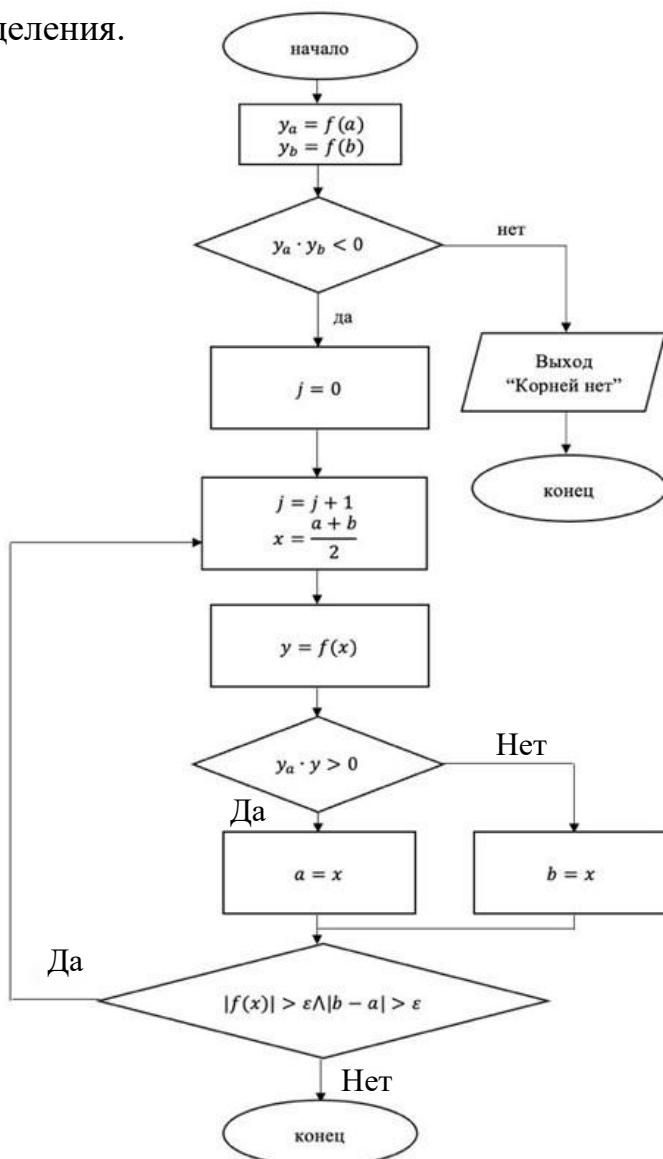


Рисунок 2 – Схема реализации метода половинного деления

Изм.	Лист	№ докум.	Подпись	Дата

УП.990000.000

Лист

13

## **2.2 Выводы по главе**

В главе представлены схема реализации метода половинного деления для решения нелинейных уравнений. Хотя метод обладает медленной сходимостью и требует противоположных знаков функции на концах интервала, он остается полезным благодаря простоте, надежности и отсутствию необходимости вычислять производные. Это делает его востребованным в задачах, где важна гарантированная сходимость, а не скорость.

Изм.	Лист	№ докум.	Подпись	Дата	Лист 14
					УП.990000.000

### **3 Программное конструирование**

В рамках данного раздела приведено обоснование выбора языка программирования и среды разработки. Представлено описание программного средства, его классов и методов.

#### **3.1 Выбор языка программирования и среды разработки**

Для реализации программного решения нелинейных уравнений методом половинного деления выбран язык программирования Python в сочетании со средой разработки PyCharm.

PyCharm — разработанная JetBrains интегрированная среда разработки является мощным инструментом для программирования на Python. Она включает поддержку важнейших научных библиотек (SymPy, NumPy, Matplotlib), используемых при выполнении вычислительных задач. Среда предлагает разработчикам продвинутые возможности: умное редактирование кода, средства визуализации результатов расчетов и комплексную отладку приложений. Эти характеристики позволяют считать PyCharm одной из лучших IDE для профессиональной разработки на Python [1].

Python — это современный интерпретируемый язык программирования высокого уровня, реализующий объектно-ориентированную парадигму. Его характерными особенностями являются динамическая типизация и автоматическое управление памятью. Язык разработан с акцентом на повышение эффективности разработки, обеспечивая исключительную читаемость кода и полную кроссплатформенную совместимость. Благодаря элегантному и интуитивно понятному синтаксису Python получил широкое распространение в различных областях, включая научные вычисления, обработку данных, машинное обучение и веб-разработку, что делает его оптимальным выбором как для начинающих, так и для опытных программистов [2].

Основные преимущества выбора PyCharm и Python

Изм.	Лист	№ докум.	Подпись	Дата	УП.990000.000	Лист 15
------	------	----------	---------	------	---------------	------------

1. PyCharm предлагает полнофункциональную IDE для Python-разработки, оснащенную интеллектуальными инструментами, включая подсветку синтаксиса, автодополнение кода, встроенный отладчик и интеграцию с системами контроля версий. Эти возможности значительно ускоряют процесс разработки и повышают продуктивность программистов.
2. Python обладает обширной стандартной библиотекой и многочисленными сторонними пакетами, что позволяет разработчикам использовать готовые решения для различных задач. Это сокращает время разработки и уменьшает объем необходимого кода.
3. Язык отличается простым и понятным синтаксисом, что делает его доступным для начинающих программистов. Высокая читаемость кода облегчает его понимание и модификацию.
4. Python и PyCharm работают на всех основных операционных системах (Windows, macOS, Linux), что позволяет создавать приложения, не зависящие от конкретной платформы, без необходимости адаптации кода.
5. Python поддерживается большим сообществом разработчиков, где можно оперативно получить помощь, найти готовые решения и быть в курсе последних тенденций в разработке.
6. PyCharm обеспечивает seamless-интеграцию с популярными инструментами разработки, включая системы контроля версий (Git), виртуальные окружения и системы тестирования, что делает процесс разработки более эффективным.

Выбор Python и PyCharm для реализации метода половинного деления обусловлен их ключевыми преимуществами: кроссплатформенностью, богатой экосистемой библиотек, простым синтаксисом и мощными инструментами разработки. Это сочетание делает их идеальным решением для задач вычислительной математики, обеспечивая эффективную реализацию

Изм.	Лист	№ докум.	Подпись	Дата
------	------	----------	---------	------

**УП.990000.000**

Лист

16

численных методов. Благодаря поддержке сообщества и удобной среде разработки, данный технологический стек позволяет создавать надежные и производительные приложения для научных вычислений.

### 3.2 Описание методов программного средства

Исходный код программы содержит два класса и методы, которые реализуют интерфейс, а также метод половинного деления для решения нелинейных уравнений.

- *bisection\_method* – основная функция, управляющая процессом решения уравнения. Она запрашивает у пользователя функцию, способ задания интервала (ручной или автоматический) и точность вычислений. Функция координирует работу всех остальных методов: проверяет корректность ввода, выполняет численное решение методом половинного деления, выводит результаты в виде таблицы итераций и предлагает построить график функции..
- *validate\_input* – метод проверки корректности введенных пользователем данных. Он проверяет, что функция является математически корректным выражением, границы интервала - числами, причем левая граница меньше правой, а точность - положительным числом. При обнаружении ошибок возвращает соответствующие сообщения.
- *draw\_function\_plot* – метод для построения графика заданной пользователем функции. На графике отображаются сама функция, найденный корень (отмеченный красной точкой), начальные границы отрезка (зеленые точки) и оси координат. Метод автоматически определяет безопасный диапазон значений x, исключая точки, где функция не определена (например, для тригонометрических функций), и фильтрует некорректные значения.

Метод половинного деления реализован в основной функции и выполняет численное решение уравнения. На каждом шаге определяется середина текущего интервала, вычисляется значение функции в этой точке,

Изм.	Лист	№ докум.	Подпись	Дата
------	------	----------	---------	------

УП.990000.000

Лист

17

после чего границы отрезка корректируются в зависимости от знака функции. Процесс повторяется, пока не будет достигнута заданная точность, не будет найдено практически точное решение или не будет превышено максимальное число итераций.

Созданный консольный интерфейс предоставляет пользователю возможность ввода исходных данных (функции, интервала и точности) с возможностью как ручного задания интервала, так и автоматического поиска подходящего отрезка. Программа выводит шаги итерационного процесса в таблицу и, по желанию пользователя, отображает график функции с отметкой найденного решения, что позволяет наглядно отслеживать ход выполнения метода половинного деления и анализировать полученные результаты.

### 3.3 Описание программы

Созданная программа предоставляет возможность решения нелинейных уравнений с использованием метода половинного деления, сопровождая процесс визуализацией результатов.

На рисунке 3 изображен основной консольный интерфейс.

```
=====
МЕТОД ПОЛОВИННОГО ДЕЛЕНИЯ
=====
Введите функцию f(x) (например, tan(x)**3 - x + 1):
Введите левую границу a (например, -1.0):
Введите правую границу b (например, 0.5):
Введите точность ε (по умолчанию 0.00001):
```

Рисунок 3 – Основной интерфейс программного средства

В верхней части интерфейса размещены поля для ввода данных:

- $f(x)$  – аналитическое выражение функции;
- $a$  – начальная граница интервала;
- $b$  – конечная граница интервала;
- «Точность  $\epsilon$ » – параметр, задающий допустимую погрешность вычислений;

Изм.	Лист	№ докум.	Подпись	Дата
------	------	----------	---------	------

УП.990000.000

Лист

18

После ввода данных появляется таблица с итогами итераций метода половинного деления и результат как показано на рисунке 4. На рисунке 5 отображается график заданной функции.

```
=====
МЕТОД ПОЛОВИННОГО ДЕЛЕНИЯ
=====
Ведите функцию f(x) (например, tan(x)**3 - x + 1): 2*cos(x + pi/6) + x**2 - 4*x + 3
Ведите левую границу a (например, -1.0): -1
Ведите правую границу b (например, 0.5): 3
Ведите точность ε (по умолчанию 0.00001):

=====
н    a            b            c            f(c)
-----
1   -1.00000000  3.00000000  1.00000000  9.43600604e-02
2   1.00000000  3.00000000  2.00000000  -2.63008489e+00
3   1.00000000  2.00000000  1.50000000  -1.62497456e+00
4   1.00000000  1.50000000  1.25000000  -8.40330267e-01
5   1.00000000  1.25000000  1.12500000  -3.89822960e-01
6   1.00000000  1.12500000  1.06250000  -1.51697453e-01
7   1.00000000  1.06250000  1.03125000  -2.96296870e-02
8   1.00000000  1.03125000  1.01562500  3.21287528e-02
9   1.01562500  1.03125000  1.02343750  1.18994776e-03
10  1.02343750  1.03125000  1.02734375  -1.42348255e-02
11  1.02343750  1.02734375  1.02539062  -6.52617039e-03
12  1.02343750  1.02539062  1.02441406  -2.66904326e-03
13  1.02343750  1.02441406  1.02392578  -7.39780623e-04
14  1.02343750  1.02392578  1.02368164  2.25025365e-04
15  1.02368164  1.02392578  1.02380371  -2.57392182e-04
16  1.02368164  1.02380371  1.02374268  -1.61870463e-05
17  1.02368164  1.02374268  1.02371216  1.04418250e-04
18  1.02371216  1.02374268  1.02372742  4.41153744e-05
19  1.02372742  1.02374268  1.02373505  1.39641072e-05
20  1.02373505  1.02374268  1.02373886  -1.11148380e-06
-----
РЕЗУЛЬТАТ:
Корень: 1.02373886
Значение функции: -1.111e-06
Количество итераций: 20

Построить график? (y/n): y
```

Рисунок 4 – Отображение результатов вычислений методом половинного деления

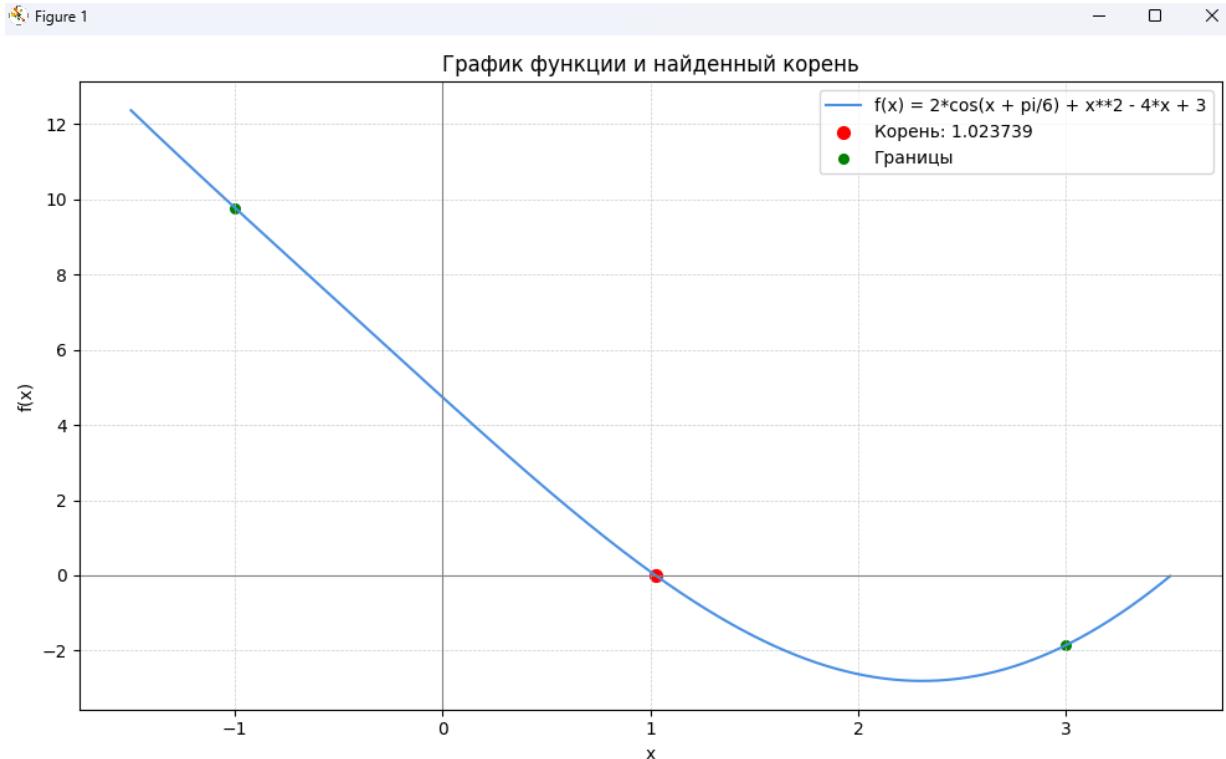


Рисунок 5 - отображение графика заданной функции

Если заданные границы интервала  $[a, b]$  не удовлетворяют условию сходимости метода половинного деления  $f(a) \cdot f(b) < 0$ , программа автоматически открывает окно ошибки с уведомлением о необходимости изменения интервала. Пример такого окна приведен на рисунке 6.

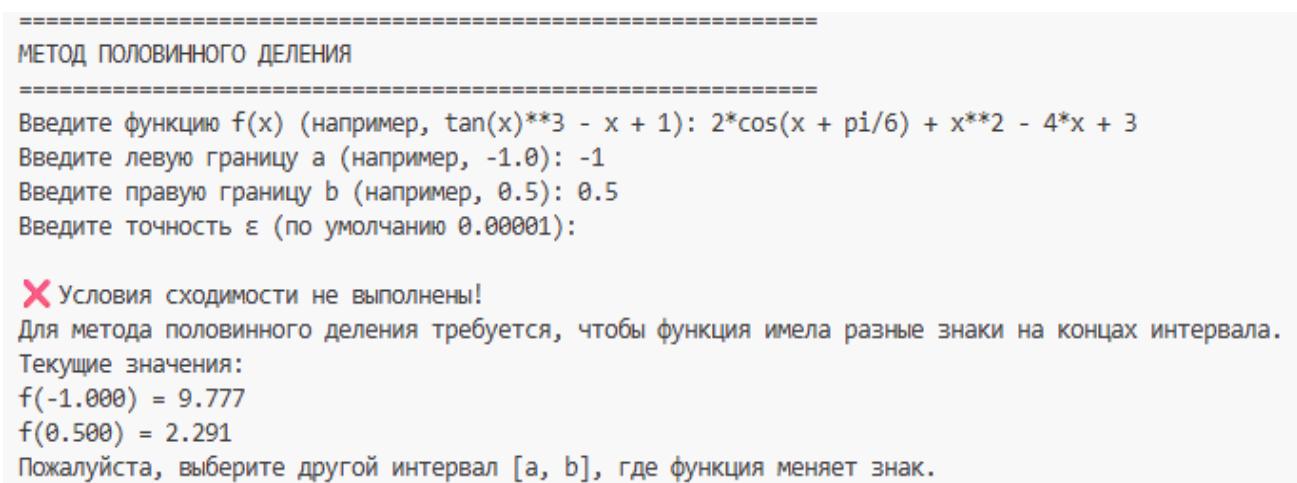


Рисунок 6 – Диалоговое окно с сообщением о несоблюдении условий сходимости

### **3.4 Выводы по главе**

В главе обоснован выбор языка программирования и среды разработки, учитывая эффективность, простоту использования и доступность библиотек для решения нелинейных уравнений. Приведено руководство по работе с программой, включая описание интерфейса, характеристику классов и основных методов. Программа позволяет численно решать нелинейные уравнения и визуализировать результаты для удобства анализа.

Изм.	Лист	№ докум.	Подпись	Дата	Лист 21
					УП.990000.000

## 4 Тестирование программного средства

В данном разделе рассмотрены примеры решения различных нелинейных уравнений с использованием метода половинного деления. Результаты вычислений представлены в табличном и графическом виде.

### 4.1 Результат работы программного средства

На рисунке 7 показан пример нелинейного уравнения варианта 29  $\arctg(x - 1) + 2 * x$ , для которого формируется график, с границами отрезка [-5,5] и точностью 0.00001.

```
=====
МЕТОД ПОЛОВИННОГО ДЕЛЕНИЯ
=====
Введите функцию f(x) (например, tan(x)**3 - x + 1): atan(x - 1) + 2*x
Введите левую границу a (например, -1.0): -5
Введите правую границу b (например, 0.5): 5
Введите точность ε (по умолчанию 0.00001): 0.00001
```

Рисунок 7 – Ввод исходных данных для нелинейного уравнения 29 варианта

При вводе последнего параметра появится таблица с результатами итераций метода половинного деления для введенного уравнения, а ниже показан построенный график, как продемонстрировано на рисунке 8.

Изм.	Лист	№ докум.	Подпись	Дата

УП.990000.000

Лист

22

n	c	f(c)
1	0.00000000000000	-0.785398163397448
2	2.50000000000000	5.982793723247329
3	1.25000000000000	2.744978663126864
4	0.62500000000000	0.891229329729428
5	0.31250000000000	0.022712653865036
6	0.15625000000000	-0.388354407884450
7	0.07812500000000	-0.184676341180762
8	0.03906250000000	-0.081456602434010
9	0.01953125000000	-0.029491852753105
10	0.00976562500000	-0.003419704972845
11	0.00488281250000	0.009638931930289
12	0.00244140625000	0.003107729827045
13	0.00122070312500	-0.000156458235173
14	0.000610351562500	0.001475518098882
15	0.0003052539062500	0.000659500511518
16	0.000152625484375000	0.000251513783580
17	0.0000763125000000	0.000047525935617
18	0.0000381562500000	-0.000054466609417
19	0.0000190781250000	-0.000003470451811
20	0.0000095373437500	0.000022027713175
21	0.0000047685371875	0.000009278623500

РЕЗУЛЬТАТ:

Корень: 0.30401707

Значение функции: 9.279e-06

Количество итераций: 21

Построить график? (y/n): y



Рисунок 8 – Таблица результатов и график

метода половинного деления 29 варианта

На рисунке 9 приведен пример ещё одного нелинейного уравнения варианта 9  $5 * \sin(x) - x + 1$ , для которого осуществляется построение графика, с границами отрезка  $[-5, 5]$  и точностью 0.00001.

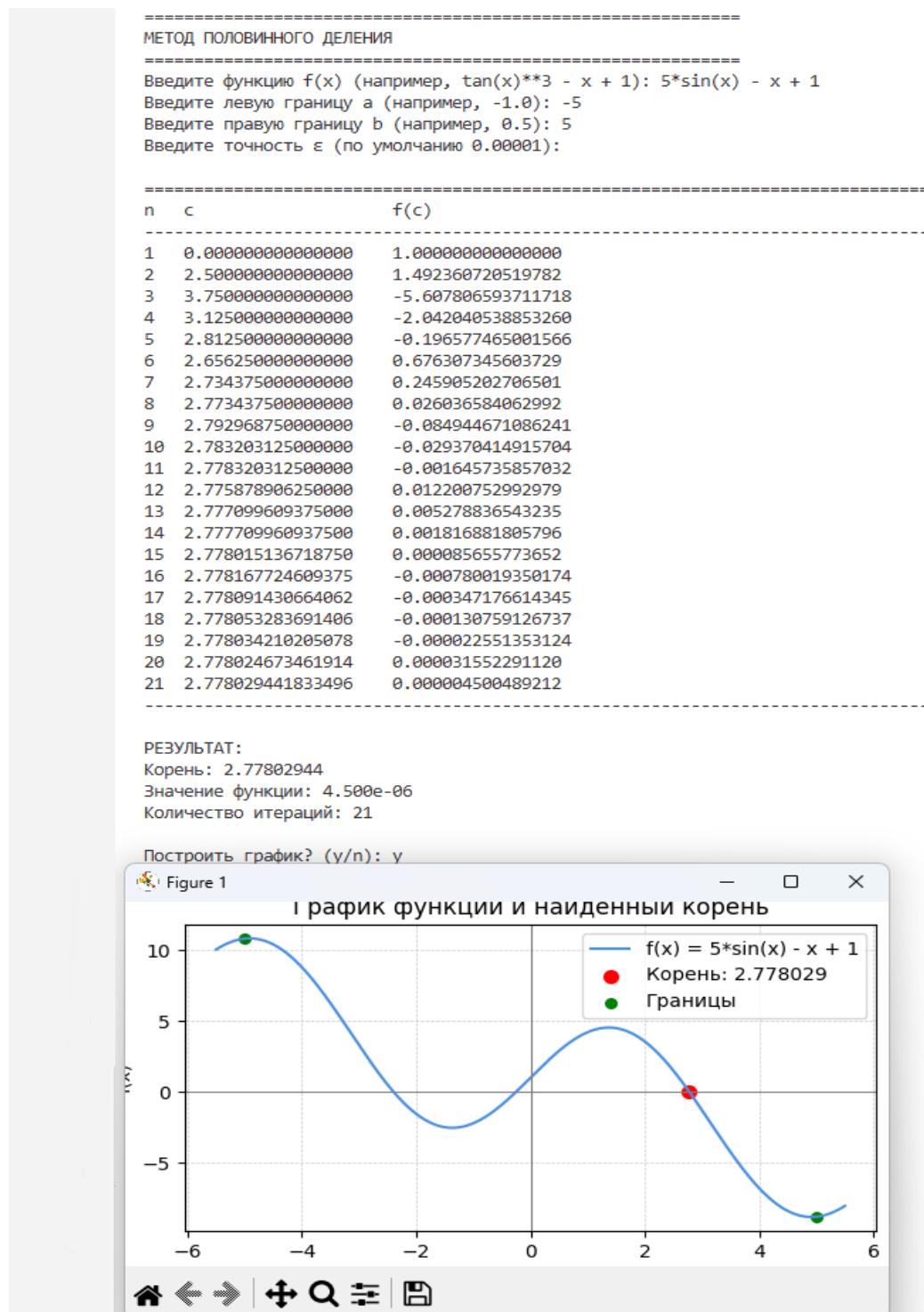


Рисунок 9 – Исходные данные, таблица результатов и график функции 10 варианта для нахождения первого корня

Изм.	Лист	№ докум.	Подпись	Дата

УП.990000.000

Лист

24

Из полученного графика видно, что на отрезках  $[-3, -2]$  и  $[-1, 0]$  есть 2 корня. На рисунке 10 приведено решение нелинейного уравнения варианта  $9 \ 5 * \sin(x) - x + 1$ , для которого осуществляется построение графика, с границами отрезка  $[-3, -2]$  и точностью  $0.00001$ .

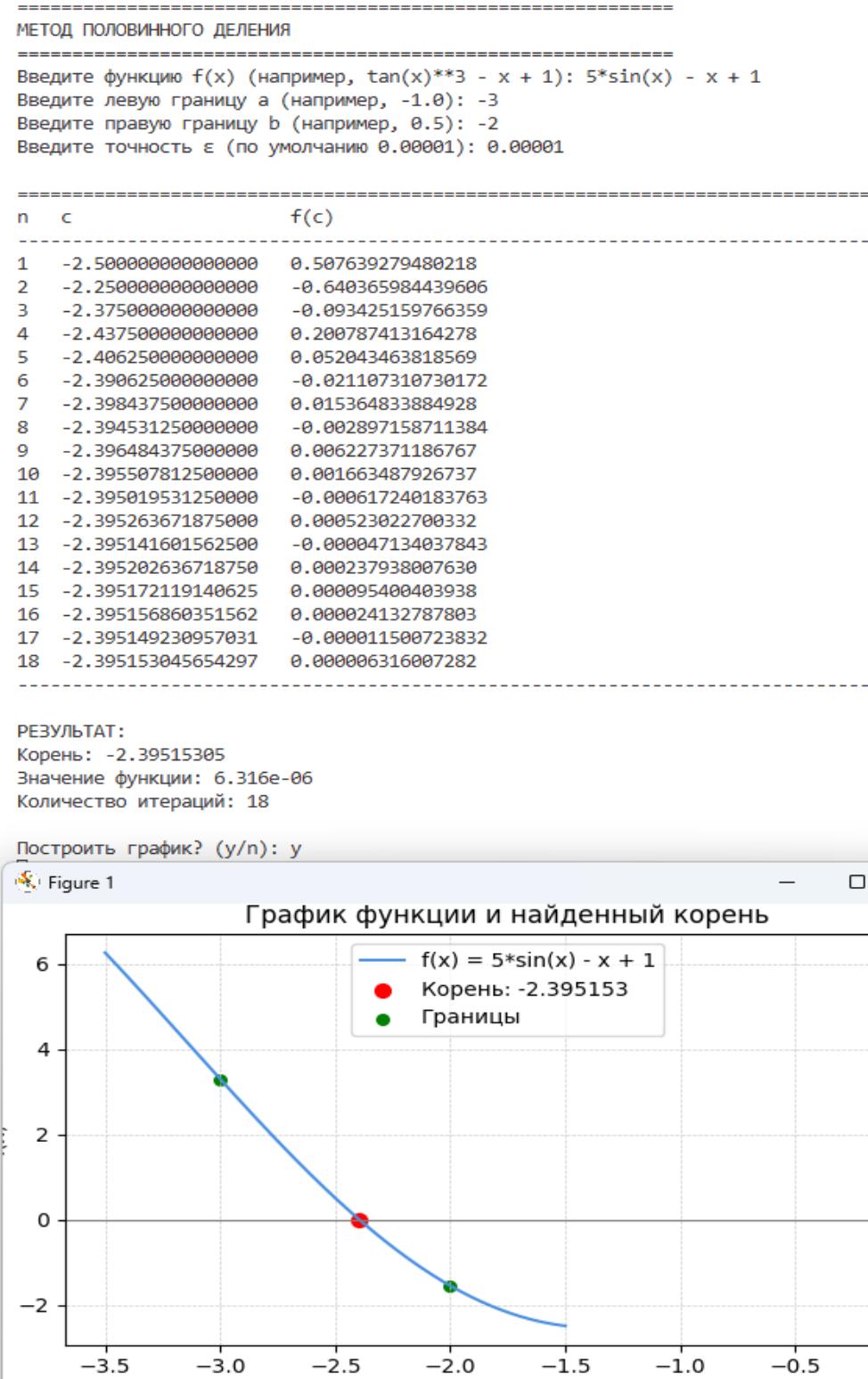


Рисунок 10 – Исходные данные, таблица результатов и график функции 10 варианта для нахождения второго корня

Изм.	Лист	№ докум.	Подпись	Дата

УП.990000.000

На рисунке 11 приведено решение нелинейного уравнения варианта 9  
 $5 * \sin(x) - x + 1 = 0$ , для которого осуществляется построение графика, с границами отрезка  $[-1, 0]$  и точностью 0.00001.

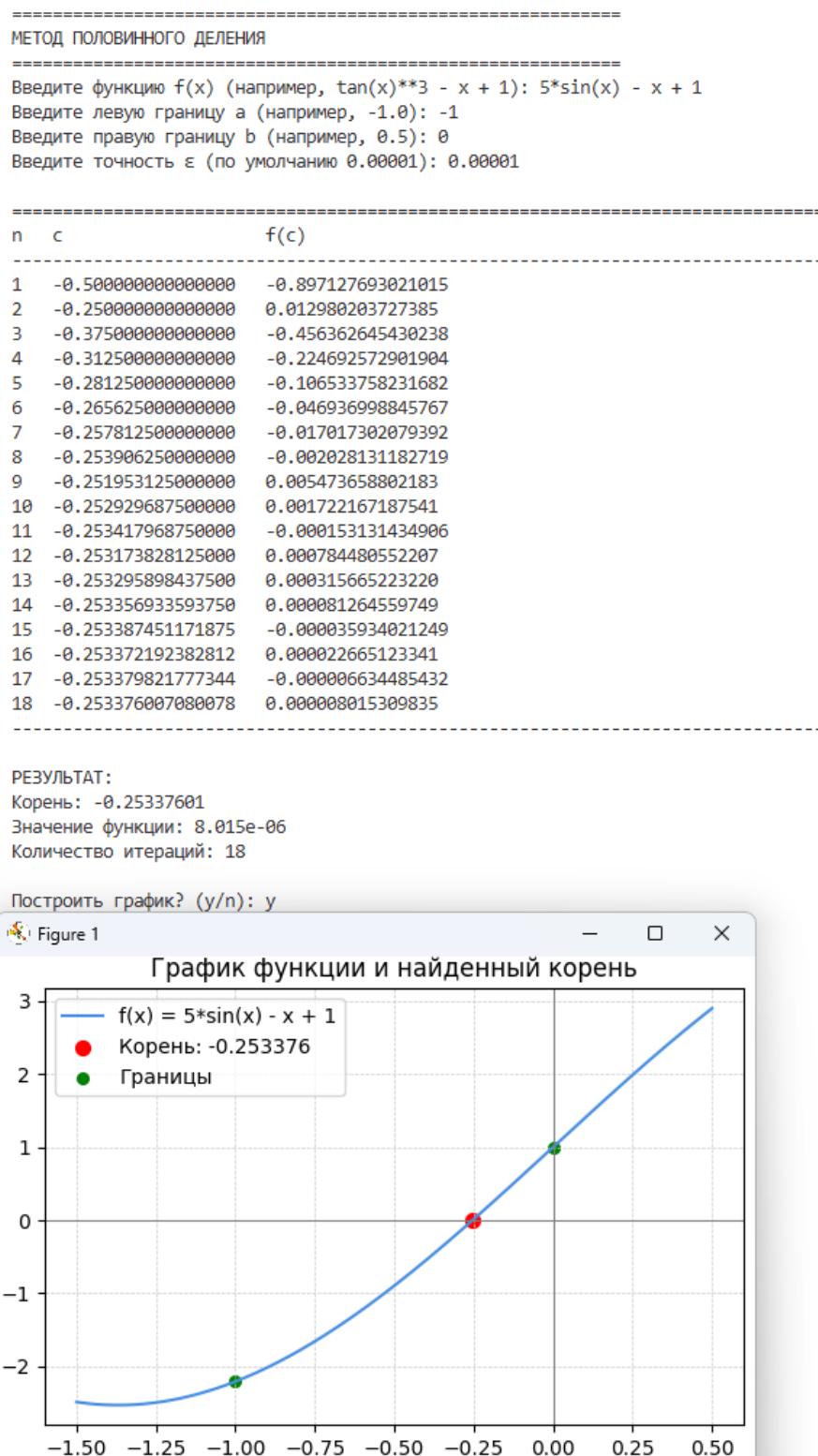


Рисунок 11 – Исходные данные, таблица результатов и график функции 10 варианта для нахождения третьего корня

Изм.	Лист	№ докум.	Подпись	Дата

УП.990000.000

Лист

26

При создании данного программного средства особое внимание уделено обеспечению его стабильности и защиты от сбоев, включая механизм обработки ошибок. В случае некорректного ввода данных пользователем отобразится окно с уведомлением об ошибке, как показано на рисунках 12, 13, 14 и 15.

```
=====
МЕТОД ПОЛОВИННОГО ДЕЛЕНИЯ
=====
Введите функцию f(x) (например, tan(x)**3 - x + 1): abracadabra
Введите левую границу a (например, -1.0): 0
Введите правую границу b (например, 0.5): 1
Введите точность ε (по умолчанию 0.00001): 0.00001

Ошибка при выполнении вычислений: cannot determine truth value of Relational: abracadabra**2 > 0
Проверьте корректность введенной функции и параметров.
```

Рисунок 12 – Ошибка при вводе функции

```
=====
МЕТОД ПОЛОВИННОГО ДЕЛЕНИЯ
=====
Введите функцию f(x) (например, tan(x)**3 - x + 1): x**4 * 3**x - 2
Введите левую границу a (например, -1.0): 1
Введите правую границу b (например, 0.5): 0
Введите точность ε (по умолчанию 0.00001): 0.00001
```

Ошибки во входных данных:

- Левая граница 'a' должна быть меньше правой границы 'b'

Рисунок 13 – Ошибка при вводе границ интервала

```
=====
МЕТОД ПОЛОВИННОГО ДЕЛЕНИЯ
=====
Введите функцию f(x) (например, tan(x)**3 - x + 1): x**4 * 3**x - 2
Введите левую границу a (например, -1.0): 1
Введите правую границу b (например, 0.5): 2
Введите точность ε (по умолчанию 0.00001): -1
```

Ошибки во входных данных:

- Точность 'ε' должна быть положительным числом

Рисунок 14 – Ошибка при вводе точности

Изм.	Лист	№ докум.	Подпись	Дата
------	------	----------	---------	------

УП.990000.000

Лист

27

=====

**МЕТОД ПОЛОВИННОГО ДЕЛЕНИЯ**

=====

Введите функцию  $f(x)$  (например,  $\tan(x)^3 - x + 1$ ):  $x^4 * 3^x - 2$   
Введите левую границу  $a$  (например,  $-1.0$ ):  $t$   
Введите правую границу  $b$  (например,  $0.5$ ):  $h$   
Введите точность  $\epsilon$  (по умолчанию  $0.00001$ ):  $k$

Ошибки во входных данных:

- Левая граница ' $a$ ' должна быть числом
- Правая граница ' $b$ ' должна быть числом
- Точность ' $\epsilon$ ' должна быть числом

Рисунок 15 – Ошибка при вводе нечисловых значений

## 4.2 Выводы по главе

Реализованное программное средство наглядно показывает работу метода половинного деления для численного решения нелинейных уравнений. Результаты вычислений отображаются в виде таблицы и графика функции.

Изм.	Лист	№ докум.	Подпись	Дата

**УП.990000.000**

Лист  
28

## **Заключение**

В ходе учебной практики изучены методы численного решения нелинейных уравнений и более подробно метод половинного деления. Разработанное программное средство реализует данный алгоритм и предлагает удобные инструменты для визуализации расчетов с использованием табличного представления данных и графиков.

Метод половинного деления, несмотря на свою простоту, требует тщательного подхода к выбору начального интервала и контроля точности. Его главное преимущество — гарантированная сходимость при выполнении условия противоположности знаков функции, что делает метод надежным инструментом даже для функций с нестандартным поведением.

Скорость сходимости метода половинного деления ниже, чем у других итерационных методов, алгоритм остается востребованным благодаря устойчивости и отсутствию необходимости вычислять производные. Дальнейшее развитие может включать его гибридизацию с другими методами для ускорения вычислений при сохранении надежности.

Разработанное программное средство, может быть использовано в учебном процессе при изучении численных методов и вычислительной математики. Оно подходит для решения разнообразных прикладных задач, связанных с поиском корней нелинейных уравнений с использованием метода половинного деления.

Изм.	Лист	№ докум.	Подпись	Дата	Лист	УП.990000.000	29

## **Перечень использованных информационных ресурсов**

1. Бахвалов Н.С., Жидков Н.П., Кобельков Г.М. Численные методы. — 6-е изд. — М.: БИНОМ. Лаборатория знаний, 2011. — 636 с.
2. Калиткин Н.Н. Численные методы. — 2-е изд. — СПб.: БХВ-Петербург, 2011. — 592 с.
3. Демидович Б.П., Марон И.А. Основы вычислительной математики. — М.: Лань, 2009. — 672 с.
4. Лутц М. Изучаем Python. — 5-е изд. — СПб.: Символ-Плюс, 2022. — 992 с.
5. Ромальо Л. Fluent Python. — 2-е изд. — СПб.: Питер, 2022. — 912 с.
6. Демидович Б.П., Марон И.А. Основы вычислительной математики. — Москва: Лань, 2006. — 672 с.
7. Саммерфилд М. Python на практике. — СПб.: Символ-Плюс, 2014. — 512 с.
8. Численные методы в Python: руководство для начинающих [Электронный ресурс] // URL: <https://www.example.com/numerical-methods-python> (дата обращения: 30.06.2025).
9. Курс лекций по численным методам [Электронный ресурс] // URL: <https://www.example.com/numerical-lectures> (дата обращения: 30.06.2025).
10. Чан У. Введение в программирование на Python [Электронный ресурс] // URL: <https://www.example.com/python-intro> (дата обращения: 30.06.2025).

Изм.	Лист	№ докум.	Подпись	Дата

**УП.990000.000**

Лист  
30

## Приложение А Исходный код программного средства

Листинг А.1 – Исходный код файла main.py:

```
from sympy import symbols, sympify, lambdify
import numpy as np
import matplotlib.pyplot as plt

def validate_input(func_str, a_str, b_str, eps_str):
    errors = []
    if not func_str.strip():
        errors.append("Функция не введена")
    else:
        try:
            x = symbols("x")
            sympify(func_str)
        except Exception as e:
            errors.append(f"Ошибка в функции: {str(e)}")

        try:
            a_val = float(a_str)
        except ValueError:
            errors.append("Левая граница 'a' должна быть числом")

        try:
            b_val = float(b_str)
        except ValueError:
            errors.append("Правая граница 'b' должна быть числом")

    if len(errors) == 0 and a_val >= b_val:
        errors.append("Левая граница 'a' должна быть меньше правой границы 'b'")
```

Изм.	Лист	№ докум.	Подпись	Дата
------	------	----------	---------	------

УП.990000.000

Лист

31

```

try:
    eps = float(eps_str)
    if eps <= 0:
        errors.append("Точность 'ε' должна быть положительным числом")
except ValueError:
    errors.append("Точность 'ε' должна быть числом")

return errors

def draw_function_plot(func_str, root, a, b):
    try:
        x = symbols("x")
        f_expr = sympify(func_str)
        f = lambdify(x, f_expr, modules=["numpy"])

        x_min = min(a, b, root) - 0.5
        x_max = max(a, b, root) + 0.5
        x_vals = np.linspace(x_min, x_max, 1000)

        with np.errstate(all="ignore"):
            y_vals = f(x_vals)
            valid_mask = np.isfinite(y_vals)
            x_vals = x_vals[valid_mask]
            y_vals = y_vals[valid_mask]

        plt.figure(figsize=(10, 6))
        plt.plot(x_vals, y_vals, color="#4a90e2", label=f"f(x) = {func_str}")
        plt.axhline(0, color="gray", linewidth=0.8)
        plt.axvline(0, color="gray", linewidth=0.8)
        plt.scatter([root], [0], color="red", s=50,
label=f"Корень: {root:.6f}")

    
```

Изм.	Лист	№ докум.	Подпись	Дата

**УП.990000.000**

Лист

32

```

        plt.scatter([a, b], [f(a), f(b)], color="green", s=30,
label="Границы")

plt.grid(True, color="#cccccc", linestyle="--",
linewidth=0.5)

plt.legend()

plt.gca().set_facecolor("#ffffff")

plt.title("График функции и найденный корень")

plt.xlabel("x")

plt.ylabel("f(x)")

plt.tight_layout()

plt.show()

except Exception as e:

    print(f"Не удалось построить график функции: {str(e)}")

    print("Возможно, функция не определена на всем интервале
или имеет особенности.")

def bisection_method():

    print("=" * 60)

    print("МЕТОД ПОЛОВИННОГО ДЕЛЕНИЯ")

    print("=" * 60)

    func_str = input("Введите функцию f(x) (например, tan(x)**3
- x + 1): ").strip()

    a_str = input("Введите левую границу a (например, -1.0):
").strip()

    b_str = input("Введите правую границу b (например, 0.5):
").strip()

    eps_str = input("Введите точность ε (по умолчанию 0.00001):
").strip()

    if not eps_str:

        eps_str = "0.00001"

```

Изм.	Лист	№ докум.	Подпись	Дата
------	------	----------	---------	------

**УП.990000.000**

```

errors = validate_input(func_str, a_str, b_str, eps_str)

if errors:

    print("\nОшибки во входных данных:")
    for error in errors:
        print(f"• {error}")

    return


try:

    a_val = float(a_str)
    b_val = float(b_str)
    eps = float(eps_str)

    x = symbols("x")
    f_expr = sympify(func_str)
    f = lambdify(x, f_expr, modules=["numpy"])

    fa = f(a_val)
    fb = f(b_val)

    if fa * fb > 0:

        print("\nX Условия сходимости не выполнены!")
        print(f"Для метода половинного деления требуется, чтобы функция имела разные знаки на концах интервала.")

        print(f"Текущие значения:")
        print(f"f({a_val:.3f}) = {fa:.3f}")
        print(f"f({b_val:.3f}) = {fb:.3f}")
        print("Пожалуйста, выберите другой интервал [a, b], где функция меняет знак.")

    return

a = a_val

```

Изм.	Лист	№ докум.	Подпись	Дата

**УП.990000.000**

Лист

34

```

b = b_val
iteration = 0
max_iterations = 1000

print("\n" + "=" * 80)
print(f"{'n':<3} {'c':<20} {'f(c)':<20}")
print("-" * 80)

while iteration < max_iterations:
    iteration += 1
    c = (a + b) / 2
    fc = f(c)

    print(f"iteration:<3} {c:<20.15f} {fc:<20.15f}")

    if abs(b - a) <= eps:
        break

    if f(a) * fc > 0:
        a = c
    else:
        b = c

root = (a + b) / 2
print("-" * 80)

if iteration == max_iterations:
    print("Предупреждение: Достигнуто максимальное число
итераций. Решение может быть неточным.")

print(f"\nРЕЗУЛЬТАТ:")
print(f"Корень: {root:.8f}")

```

Изм.	Лист	№ докум.	Подпись	Дата
------	------	----------	---------	------

**УП.990000.000**

```
print(f"Значение функции: {f(root):.3e}")  
print(f"Количество итераций: {iteration}")  
  
plot_choice = input("\nПостроить график? (y/n):")  
").strip().lower()  
  
if plot_choice in ['y', 'yes', 'д', 'да']:  
    draw_function_plot(func_str, root, a_val, b_val)  
  
except Exception as e:  
    print(f"\nОшибка при выполнении вычислений: {str(e)}")  
    print("Проверьте корректность введенной функции и  
параметров.")  
  
if __name__ == "__main__":  
    bisection_method()
```

Изм.	Лист	№ докум.	Подпись	Дата

УП.990000.000

Лист

36