***Федеральное государственное бюджетное образовательное учреждение высшего образования***

|  |  |
| --- | --- |
|  | ***«Московский государственный технический университет  имени Н.Э. Баумана»***  ***(национальный исследовательский университет)***  ***(МГТУ им. Н.Э. Баумана)*** |

ФАКУЛЬТЕТ \_\_\_\_ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ\_\_\_\_\_\_\_\_

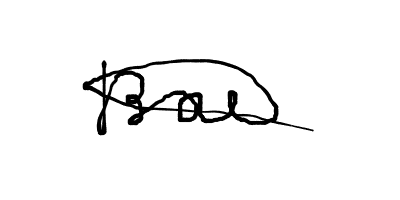
КАФЕДРА \_\_\_\_\_\_Компьютерные Системы и сети (ИУ6)\_\_\_\_\_\_\_\_\_\_\_

**Отчет**

**по лабораторной работе № 3**

**Название лабораторной работы: C#. Создание консольного приложения**

**Дисциплина: Алгоритмизация и программирование**



Студент гр. ИУ6-25Б **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** В.А. Бирюков

(Подпись, дата) (И.О. Фамилия)



Преподаватель  **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**  А.А. Веселовский

(Подпись, дата) (И.О. Фамилия)

Москва, 2025

**Цели**: научиться работать в среде разработки консольных приложений на языке программирования C#, научиться работать с средствами отладки

**Часть 1.** Изучить среду разработки консольных приложений на языке программирования C#.

В качестве примера использовать приложение, определяющее корни квадратного уравнения ax2+bx+c = 0. Предусмотреть возможность отсутствия корней при введенных пользователем коэффициентах, предусмотреть исключения при вводе нечисловых данных. Отладить и протестировать программу.

Напишем программу на языке C#

class Program {

static void Main() {

double a = 0, b = 0, c = 0, x1, x2, discr;

bool Flag = true;

while (Flag)

{

try

{

Console.WriteLine("Введите коэффициент перед x^2:");

a = double.Parse(Console.ReadLine());

if (a == 0)

{

Console.WriteLine("Это не квадратное уравнение. Коэффициент a не может быть равен 0.");

}

Console.WriteLine("Введите коэффициент перед x^1:");

b = double.Parse(Console.ReadLine());

Console.WriteLine("Введите коэффициент перед x^0:");

c = double.Parse(Console.ReadLine());

Flag = false;

}

catch (FormatException)

{

Console.WriteLine("Ошибка. Введите числовые значения!");

}

}

discr = b \* b - 4 \* a \* c;

if (discr > 0)

{

x1 = (-b - Math.Sqrt(discr)) / (2 \* a);

x2 = (-b + Math.Sqrt(discr)) / (2 \* a);

Console.WriteLine($"Первый корень: {x1}\nВторой корень: {x2}");

}

else if (discr == 0)

{

x1 = -b / (2 \* a);

Console.WriteLine($"Единственный корень: {x1}");

}

else

{

Console.WriteLine("Действительных корней нет.");

}

}

}

Протестируем программу на разных введённых данных

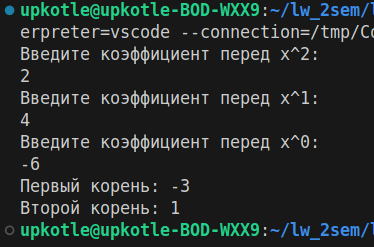


Рисунок 1 – Результат работы программы в стандартном случае

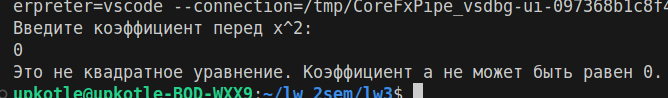


Рисунок 2 – Результат работы программы при a = 0

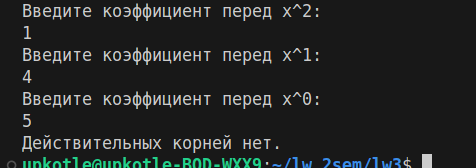
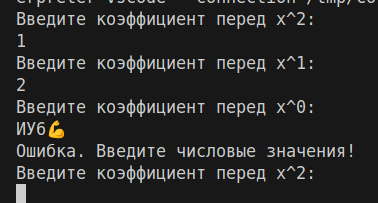


Рисунок 3 – Результат работы программы при отсутствии действительных корней

Рисунок 4 – Обработка исключений

Программа соответсвует всем условиям из задачи, успешно обрабатывает случаи с отсутствием действительных корней и с вводом нечисловых значений

Составим схему алгоритма и диаграмму классов для этой программы

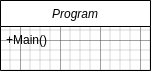


Рисунок 5 – диаграмма классов

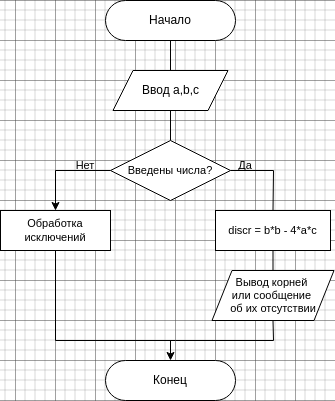
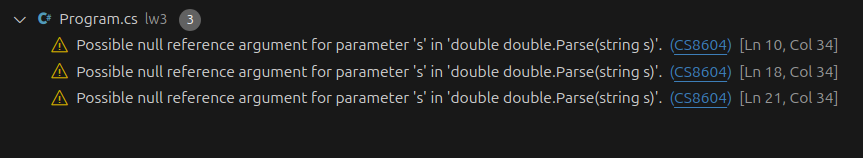


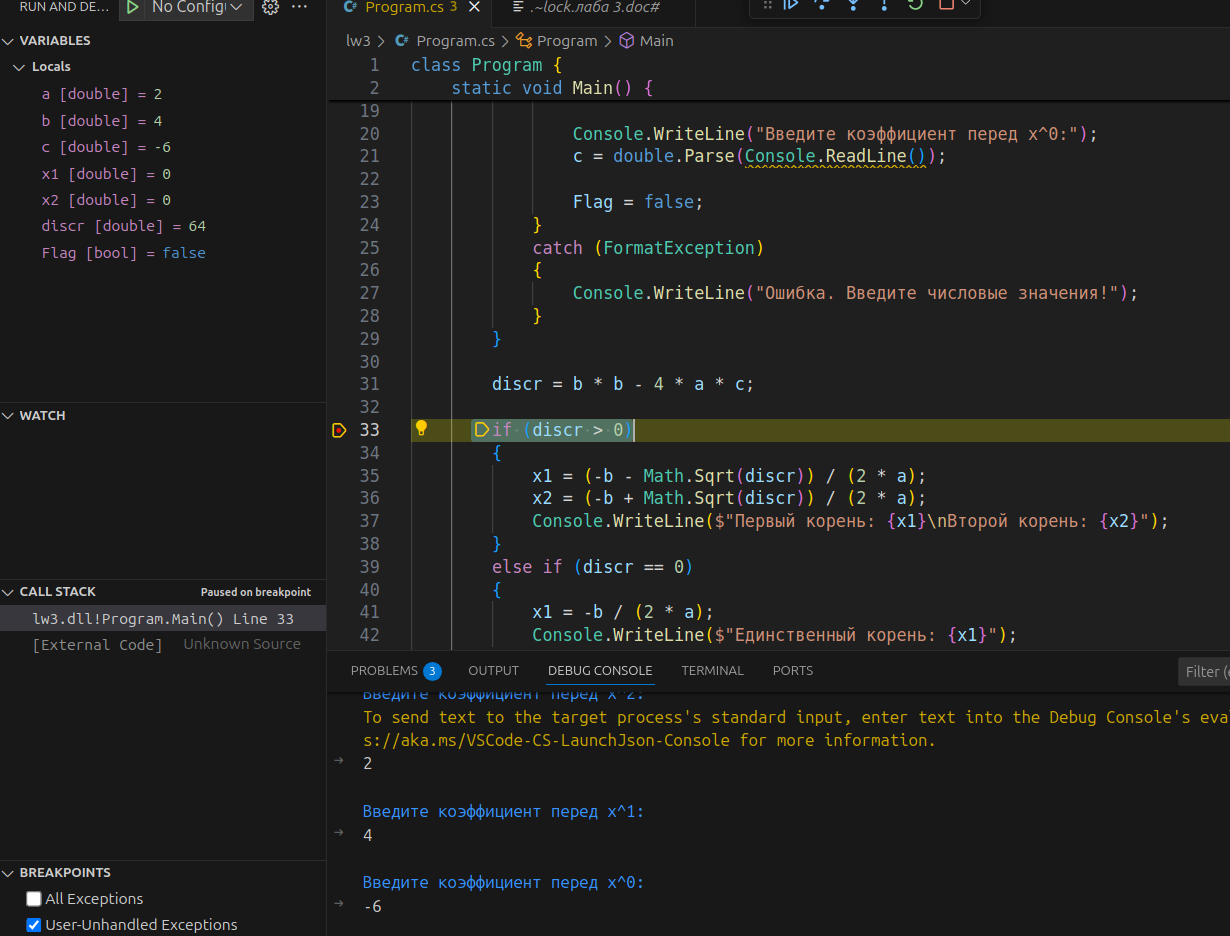
Рисунок 6 – схема алгоритма программы

**Часть 2.** Изучить диагностические сообщения компилятора и средства отладки. Привести в отчете примеры диагностических сообщений, а также скриншоты результатов применения средств отладки для установки контрольных точек и просмотра промежуточных результатов.

Рисунок 7 — Диагностические сообщения компилятора

Видим, что компилятор проанализировал код и **не может гарантировать**, что результат Console.ReadLine() (или другой переменной, передаваемой в double.Parse) **не равен** null.

А метод double.Parse(string s) не принимает null, иначе будет выброшено исключение ArgumentNullException.

Рисунок 8 — Отладка программы

Поставим точку останова на проверке положительности дискриминанта и посмотрим на промежуточные результаты. Средства отладки позволяют увидеть значения переменных в конкретный момент.

**Вывод:** научился работать в среде разработки консольных приложений на языке программирования C#, научился работать с средствами отладки.