Министерство цифрового развития, связи и массовых коммуникаций Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Санкт-петербургский государственный университет телекоммуникаций им. проф. М. А. Бонч-Бруевича»

Факультет иксс

кафедра пиИвт

(СПбГУТ)

ОтчЁт  
ЛАБОРАТОРНАЯ РАБОТА №1

**Автоматизация решения квадратного уравнения**

Руководитель,  
старший преподаватель Ерофеев С. А.

подпись, дата

Исполнитель,  
группа ИКПИ-33 Коньков М. Д.

подпись, дата

Постановка задачи

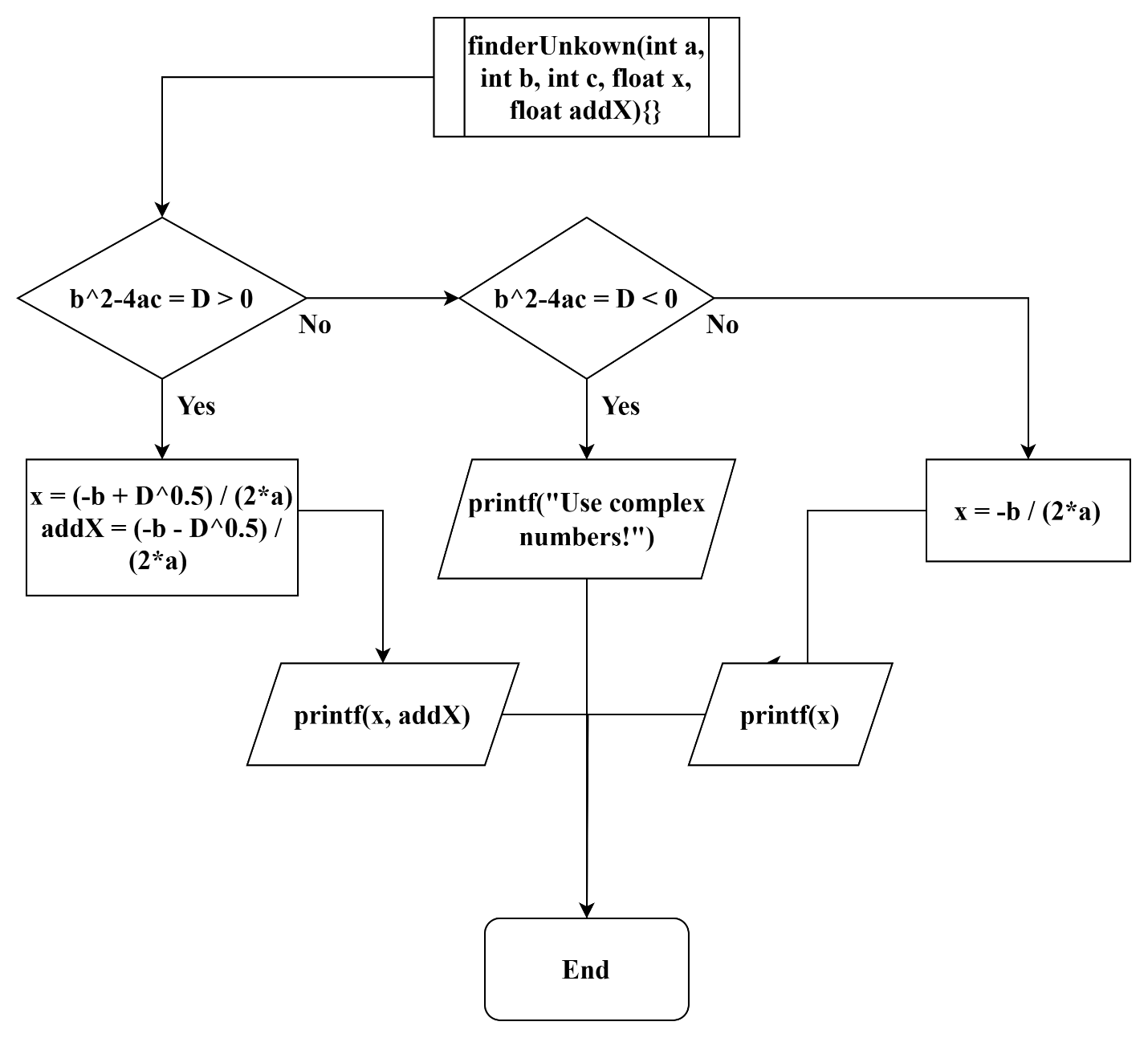
**Общая формулировка:**

Необходимо написать программу, которая будет высчитывать значения неизвестных в квадратном уравнении, где коэффициенты задаёт пользователь. Должны быть учтены все случаи уравнений в зависимости от переменных.

**Процесс выполнения лабораторной работы можно разбить на подзадачи:**

1. Подключаем необходимые заголовочные файлы (functions.h и functions.c в том числе, они помогают вынести дополнительные функции в отдельный файл), объявляем переменные для дальнейших вычислений и, что опционально, открываем цикл повторения всей программы. Также очищаем экран при попытке пользователя запустить программу;
2. Просим пользователя ввести три переменные (a, b, c), называемые коэффициентами при неизвестных, после чего выводим полученное уравнение в отчётливом математическом квадратном виде;
3. Вызываем функцию из отдельного, ранее подключённого, заголовочного файла (см. шаг 1). В эту функцию передаём коэффициенты и две переменные для будущих неизвестных (х) в качестве параметров;
4. Внутри функции описан массивный условный конструкт для учёта всех ситуаций (зависят от ранее введённых переменных) в соответствие с математическими правилами. Здесь расписываем условия (фактически их 8), после чего возвращаем 0.
5. После выполнения нужной операции, вывода ответа, программа спрашивает пользователя о том, есть ли необходимость повторить выполнение. При соблюдении условия программа переходит к следующий итерации до тех пор, пока пользователь не захочет выйти.

## Общий алгоритм решения:



**Переменные и их спецификация:**

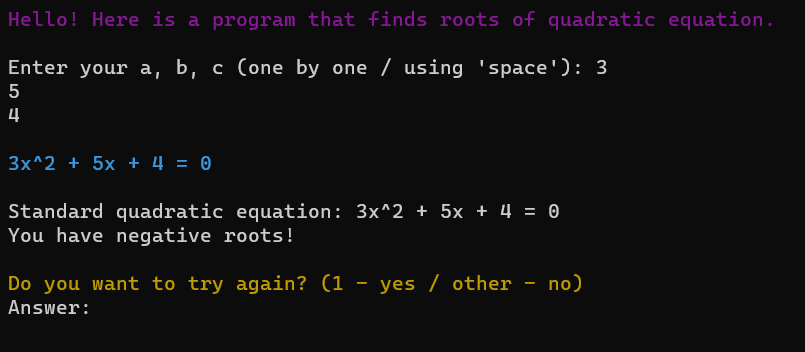
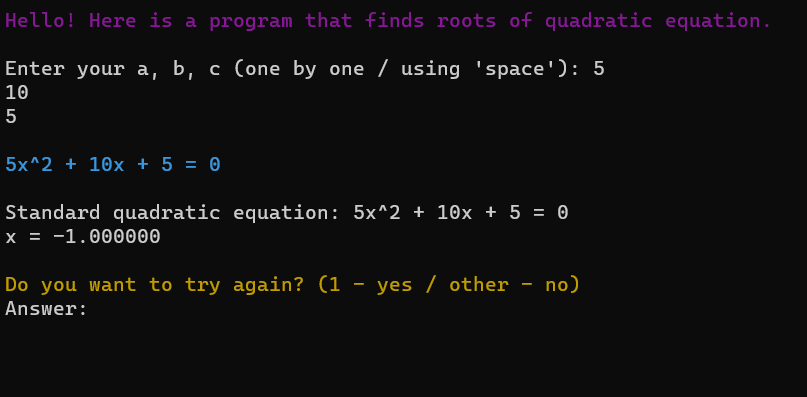
|  |  |  |  |
| --- | --- | --- | --- |
| N | Обозначение в задаче | Идентификатор и тип | Назначение |
| 1 | *-* | *char again* | Повторение |
| 2 | *a, b, c* | *int n* | Коэффициенты при х |
| 3 | *x1, x2* | *float x, addX* | Неизвестные |
| 4 | *D* | *int D* | Дискриминант |

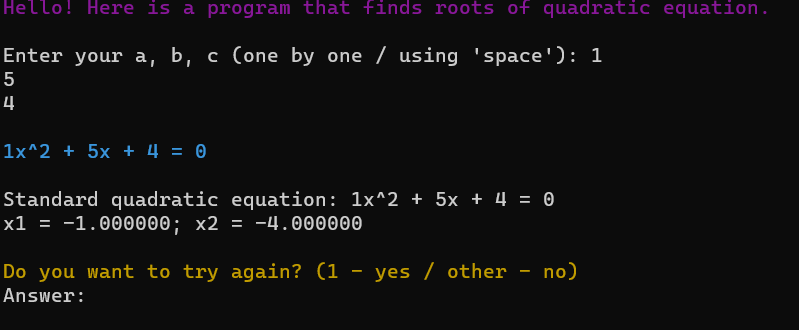
**Int**: диапазон чисел от -2 147 483 648 до 2 147 483 647;

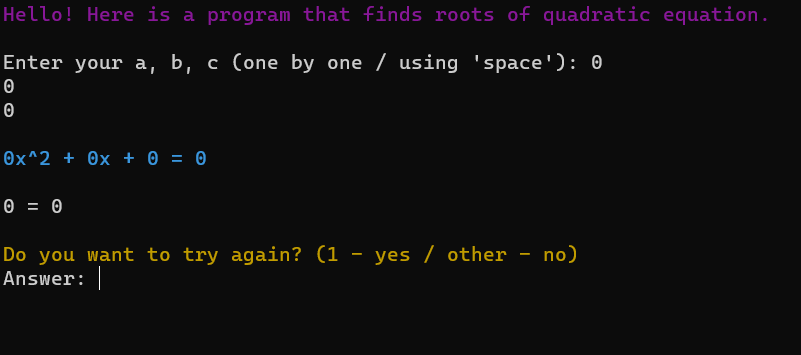
**Float**: диапазон чисел от от -3.4\*1038 до 3.4\*1038;

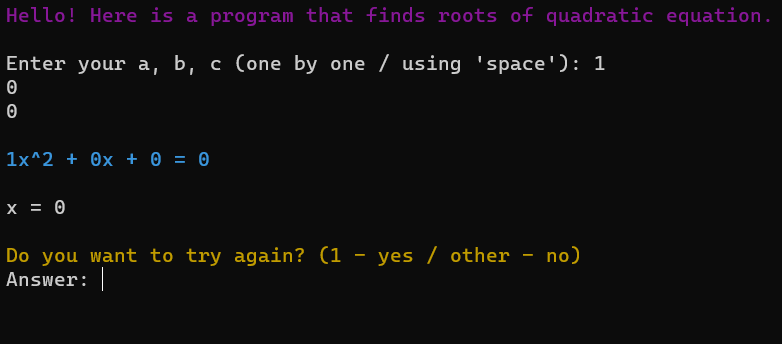
**Char**: Все знаковые значения char находятся в диапазоне от –128 до 127. Все беззнаковые значения char находятся в диапазоне от 0 до 255.

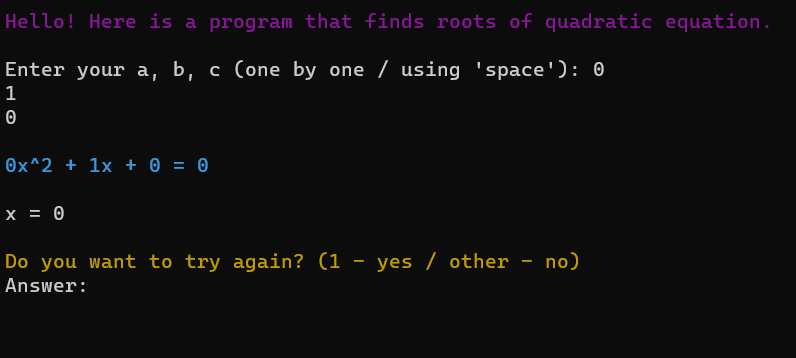
## Тестирование:

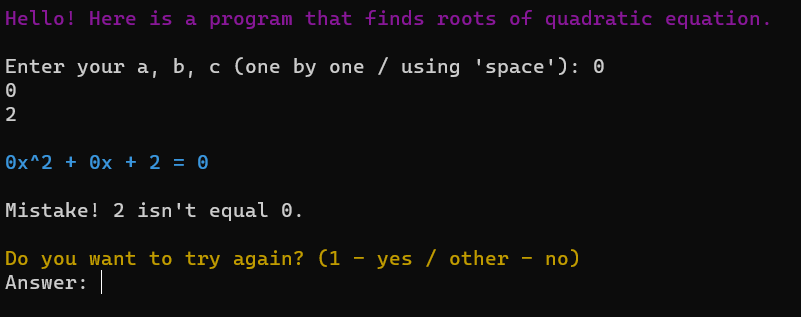


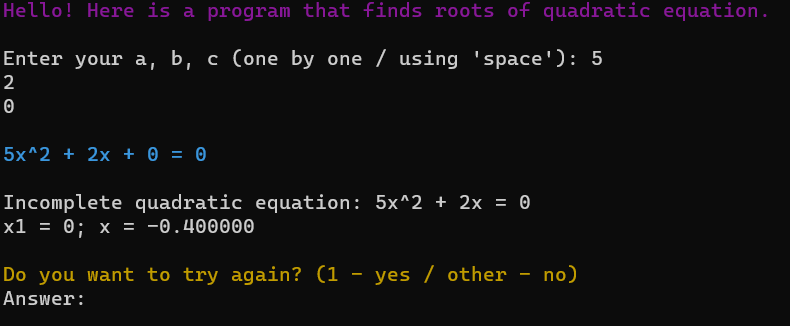


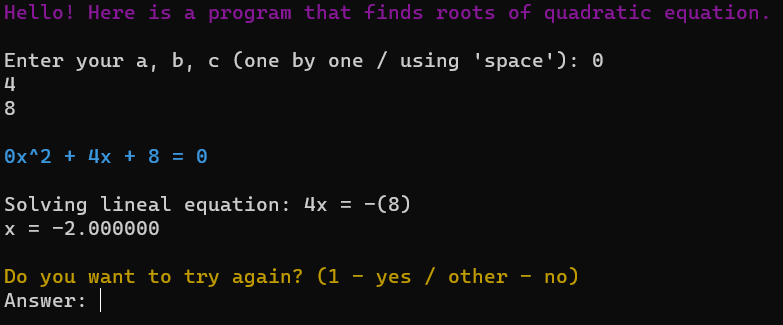












Выше показаны примеры всех выводов программы (всего 9 случаев).

1. Листинг программы AE\_Lab1.c (C)

#include "stdio.h"

#include "stdlib.h"

#include "math.h"

#include "functions.c"

int main()

{

int again; // checking-for-repeat var

int a, b, c; // coefficients before x

float x, addX; // unknowns (х1, х2)

while(1)

{

system("cls");

printf("\033[0;35mHello! Here is a program that finds roots of quadratic equation.\033[0m\n\n");

printf("Enter your a, b, c (one by one / using 'space'): ");

scanf("%d %d %d", &a, &b, &c);

printf("\n\033[0;36m%dx^2 + %dx + %d = 0\033[0m\n\n", a, b, c);

finderUnkown(a, b, c, x, addX);

// Repeat-check part

printf("\n\n\033[0;33mDo you want to try again? (1 - yes / other - no)\033[0m \nAnswer: ");

scanf("%d", &again);

if (again != 1)

{

break;

}

}

return 0;

}

1. листинг файла functions.c (c)

#include "stdio.h"

#include "stdlib.h"

#include "math.h"

int finderUnkown(int a, int b, int c, float x, float addX)

{

int D; // discriminant

if (a == 0)

{

if (b == 0)

{

if (c == 0)

{

printf("x – any real number");

return 0;

}

else

{

printf(“No solution”);

return 0;

}

}

else

{

if (c == 0)

{

printf("x = 0");

}

else

{

printf("Solving lineal equation: %dx = -(%d)\n", b, c);

x = (float)(-c) / (float)(b);

printf("x = %f", x);

}

}

}

else

{

if (b == 0)

{

if (c == 0)

{

printf("x = 0");

}

else

{

printf("Incomplete quadratic equation: %dx^2 = -(%d)\n", a, c);

x = pow(-c / a, 1.0 / 2.0);

addX = -x;

printf("x1 = %f, x2 = %f", x, addX);

}

}

else

{

if (c == 0)

{

printf("Incomplete quadratic equation: %dx^2 + %dx = 0\n", a, b);

x = (float)(-b) / (float)(a);

printf("x1 = 0; x = %f", x);

}

else // Complete quadratic equation

{

printf("Standard quadratic equation: %dx^2 + %dx + %d = 0\n", a, b, c);

D = pow(b, 2) - 4 \* a \* c;

if (D > 0)

{

x = (-b + pow(D, 1.0 / 2.0)) / (2 \* a);

addX = (-b - pow(D, 1.0 / 2.0)) / (2 \* a);

printf("x1 = %f; x2 = %f", x, addX);

}

else if (D == 0)

{

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

printf("x = %f", x);

}

else

{

printf("You have negative roots!");

return 0;

}

}

}

}

return 0;

}}

1. листинг файла functions.h (C)

#ifndef functions\_h

#define functions\_h

int finderUnkown(int a, int b, int c, float x, float addX);

#endif

**Вывод:**

В ходе выполнения лабораторной работы я улучшил навык построения блок-схем, углубил их понимание. То же касается решения квадратных уравнений в целом и написания программ на языке C в среде разработки Sublime Text 4. Необходимо было учесть все условия в рамках выполнения программы по поиску корней квадратного уравнения и полученный результат вывести пользователю. Функциональные требования, поставленные перед программой были выполнены.