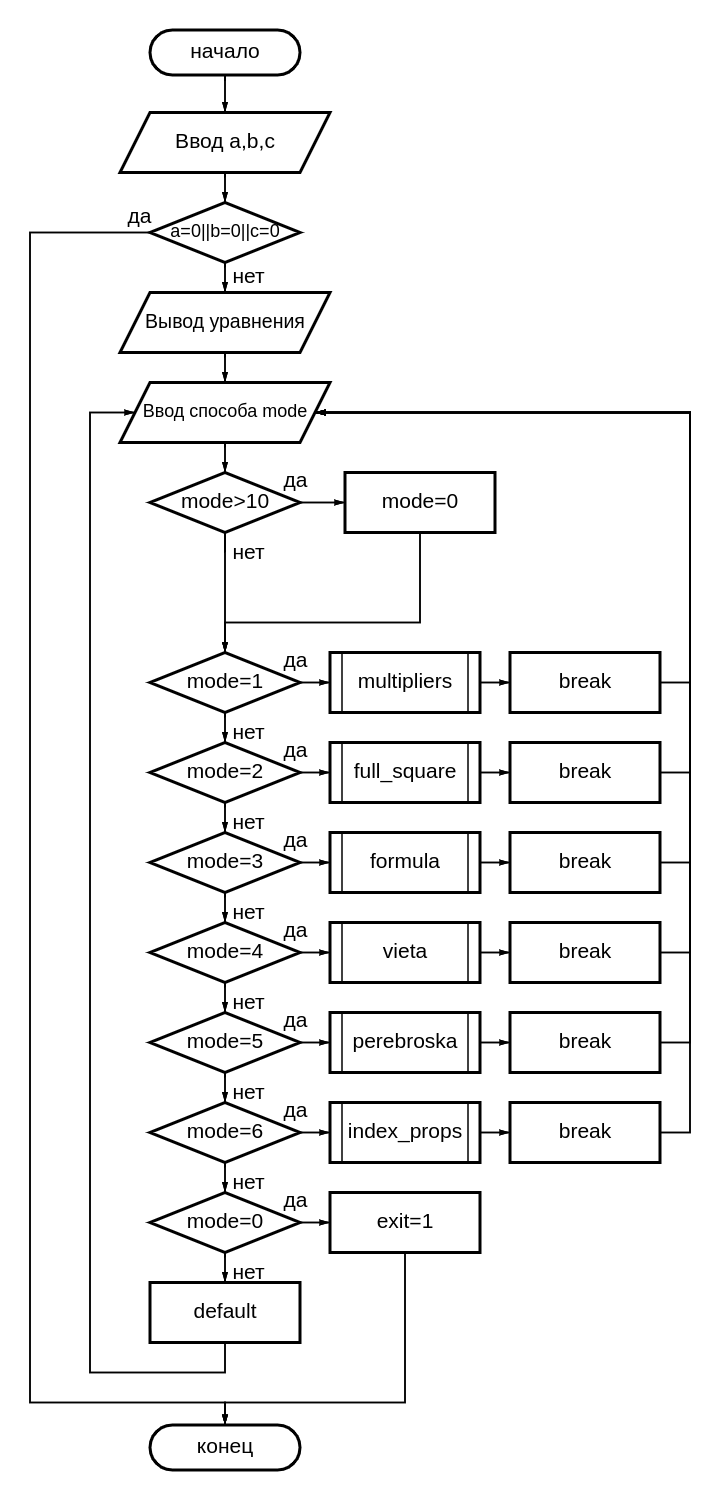
Ященков И.А. ПИБ-31з

Лабораторная работа №4. Работа с функциями

**Постановка задачи:** необходимо сделать программу решения квадратного уравнения (10 разными способами) и отчет. В программе на Си реализовать с использованием функций меню выбора способа решения и само решение. Ввод-вывод с/на экран(а). Функции-решения вынести в отдельный файл my\_func.h.

**Техническое задание:** см. ТЗ.pdf.

**Блок-схема основного алгоритма:**

****

**Листинг:**

**main.c**

#include "my\_func.h"

void main()

{

printf("\nПРОГРАММА ДЛЯ РЕШЕНИЯ КВАДРАТНЫХ УРАВНЕНИЙ\n\n\tax^2+bx+c=0\n\n");

int exit=0, a, b, c;

char input[10];

printf("Введите коэффиценты уравнения.\n\n\tВНИМАНИЕ!\nЛюбое не цифровое значение = 0!\nВещественные берутся без дробной части!\n\n");

printf("Введите коэффицент а: ");

scanf("%s[^\n]\n",input);

sscanf(input, "%d", &a);

printf("Введите коэффицент b: ");

scanf("%s[^\n]\n",input);

sscanf(input, "%d", &b);

printf("Введите коэффицент c: ");

scanf("%s[^\n]\n",input);

sscanf(input, "%d", &c);

if(a==0 || b==0 || c==0)

{

printf("Это уравнение решается без использования способов в этой программе!\n\n");

return;

}

printf("\nВаше уравнение: %dx^2%s%dx%s%d=0\n\n",a,b>0?"+":"",b,c>0?"+":"",c);

printf("Выберите способ решения:\n\n");

printf("1 - Разложение левой части уравнения на множители\n");

printf("2 - Метод выделения полного квадрата\n");

printf("3 - Решение квадратных уравнений по формуле\n");

printf("4 - Решение уравнений с использованием теоремы Виета\n");

printf("5 - Решение уравнений способом «переброски»\n");

printf("6 - Свойства коэффициентов квадратного уравнения\n");

printf("7 - Графическое решение квадратного уравнения\n");

printf("8 - Решение квадратных уравнений с помощью циркуля и линейки\n");

printf("9 - Решение квадратных уравнений с помощью номограммы\n");

printf("10 - Геометрический способ решения квадратных уравнений\n");

printf("0 || другое значение - ВЫХОД\n");

while (!exit)

{

int mode=0;

printf("\nСпособ: ");

scanf("%s[^\n]\n\n",input);

sscanf(input, "%d", &mode);

if(mode>10) mode=0;

switch (mode)

{

case 1:

multipliers(a,b,c);

break;

case 2:

full\_square(a,b,c);

break;

case 3:

formula(a,b,c);

break;

case 4:

vieta(a,b,c);

break;

case 5:

perebroska(a,b,c);

break;

case 6:

index\_props(a,b,c);

break;

case 0:

printf("Выход из программы...\n");

exit=1;

break;

default:

printf("Графические способы решения находятся в разработке!\n");

break;

}

}

}

**my\_func.h**

#pragma once

#include <stdio.h>

#include <stdlib.h>

#include <math.h>

void multipliers(int a, int b, int c)

{

int mn\_1=0, mn\_2=0, exp1=c, exp2=b;

if (a!=1)

{

if(c%a!=0 && b%a!=0)

{

printf("При делении c и b на a не получается целое число! МЕТОД В РАЗРАБОТКЕ...\n");

return;

}

printf("Разделим уравнение на а:\n");

exp1=c/a;

exp2=b/a;

printf("x^2%s%dx%s%d\n",exp2>0?"+":"",exp2,exp1>0?"+":"",exp1);

}

while(mn\_1+mn\_2!=exp2 || exp1%mn\_1!=0)

{

exp2>0?mn\_1++:mn\_1--;

mn\_2=exp1/mn\_1;

if (abs(mn\_1)>abs(exp1))

{

printf("Общие множители не найдены! (Метод работает только с целыми множителями)\n");

return;

}

}

printf("Разложим левую часть на множители:\n");

printf("x^2%s%dx%s%dx%s%d = x(x%s%d)%s%d(x%s%d) = (x%s%d)(x%s%d)\n",mn\_2>0?"+":"",mn\_2,mn\_1>0?"+":"",

mn\_1,exp1>0?"+":"",exp1,mn\_2>0?"+":"",mn\_2,mn\_1>0?"+":"",mn\_1,mn\_2>0?"+":"",

mn\_2,mn\_1>0?"+":"",mn\_1,mn\_2>0?"+":"",mn\_2);

printf("Следовательно, уравнение можно переписать как:\n");

printf("(x%s%d)(x%s%d)=0\n",mn\_1>0?"+":"",mn\_1,mn\_2>0?"+":"",mn\_2);

if (mn\_1==mn\_2)

{

printf("Значит, корень уравнения один: x=%d.\n",-mn\_1);

}

else

{

printf("Значит, корни уравнения: x1=%d; x2=%d.\n",-mn\_1,-mn\_2);

}

}

void full\_square(int a, int b, int c)

{

if (b%2!=0)

{

printf("b нельзя нацело разделить на 2! (Метод пока не работает с плавающей точкой)");

return;

}

if(a<0)

{

printf("Домножим уравнение на -1:\n");

a\*=-1;

b\*=-1;

c\*=-1;

printf("%dx^2%s%dx%s%d\n",a,b>0?"+":"",b,c>0?"+":"",c);

}

if (a!=1) printf("\tВНИМАНИЕ!\nПри a!=1 могут быть ошибки в расчетах!\n");

int mult = b/2;

int sqrt\_a=(int)sqrt(a);

printf("Выделим в левой части полный квадрат:\n");

printf("%dx^2%s%dx = %dx^2%s%d\*2\*x\n",a,b>0?"+":"",b,a,mult>0?"+":"",mult);

printf("Чтобы получить полный квадрат, нужно прибавить %d^2:\n",abs(mult));

printf("%dx^2%s%d\*2\*x+%d^2 = (x%s%d)^2\n",sqrt\_a,mult>0?"+":"",mult,abs(mult),mult>0?"+":"",mult);

printf("Преобразуем левую часть уравнения прибавляя и вычитая %d^2:\n",abs(mult));

printf("%dx^2%s%dx%s%d = (x%s%d)^2-%d%s%d = (x%s%d)^2%s%d\n",a,b>0?"+":"",b,c>0?"+":"",c,

mult>0?"+":"",mult,(int)pow(mult,2),c>0?"+":"",c,

mult>0?"+":"",mult,(-(int)pow(mult,2)+c)>=0?"+":"",

-(int)pow(mult,2)+c);

if(-(-pow(mult,2)+c)<0)

{

printf("Под знаком корня отрицательное число! Корней нет!\n");

return;

}

if(-(-pow(mult,2)+c)==0)

{

printf("Таким образом, x%s%d=0\n",mult>0?"+":"",mult);

printf("Значит, корень уравнения: x=%d.\n",-mult);

}

else

{

printf("Таким образом, x%s%d=%.0f или x%s%d=-%.0f\n",mult>0?"+":"",mult,sqrt(-(-(int)pow(mult,2)+c)),

mult>0?"+":"",mult,sqrt(-(-(int)pow(mult,2)+c)));

printf("Значит, корни уравнения: x1=%0.f; x2=%.0f.\n",sqrt(-(-(int)pow(mult,2)+c))-mult,-sqrt(-(-(int)pow(mult,2)+c))-mult);

}

}

void formula(int a, int b, int c)

{

printf("Рассчитаем дискриминант D:\n");

printf("\tВНИМАНИЕ!\nПри получении не целого квадратного корня из D могут быть ошибки в расчетах!\n");

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

if (D<0)

{

printf("Дискриминант отрицательный! Корней нет!\n");

return;

}

if (D==0)

{

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

printf("Дискриминант равен нулю! Уравнение имеет корень: x=%d\n",x);

}

else

{

float x1 = (-b + sqrt(D))/(2\*a);

float x2 = (-b - sqrt(D))/(2\*a);

printf("Дискриминант положительный! Уравнение имеет корни: x1=%.1f; x2=%.1f\n",x1,x2);

}

}

void vieta(int a, int p, int q)

{

printf("По теореме Виета корни уравнения удовлетворяют системе:\n");

printf("\tx1\*x2 = %d\n",q);

printf("\tx1 + x2 = %d\n",-p);

if (a!=1)

{

if (a!=1) printf("\tВНИМАНИЕ!\nМогут быть ошибки в расчетах, если b и c не делятся нацело на а!\n");

printf("Приведем к нужному виду, разделив уравнение на а\n");

p/=a;

q/=a;

}

int x1=0, x2=0;

if (q>0)

{

printf("Так как свободый член q>0, то уравнение имеет два одинаковых по знаку корня\n");

printf("Так как p%s0, оба корня %s\nСледовательно: ",p>0?">":"<",p>0?"отрицательны":"положительны");

while(x1\*x2!=q || x1+x2!=-p)

{

p>0?x1--:x1++;

x2=-p-x1;

if (abs(x1)>10000)

{

printf("ОШИБКА! Корни не найдены\n");

return;

}

}

}

else

{

printf("Так как свободый член q<0, то уравнение имеет два разных по знаку корня\n");

printf("Так как p%s0, больший по модулю корень %s\nСледовательно: ",p>0?">":"<",p>0?"отрицателен":"положителен");

while(x1\*x2!=q || x1+x2!=-p)

{

p>0?x1++:x1--;

x2=-p-x1;

if (abs(x1)>10000)

{

printf("ОШИБКА! Корни не найдены\n");

return;

}

}

}

if (x1==x2) printf("x=%d\n",x1);

else printf("x1=%d; x2=%d\n",x1,x2);

}

void perebroska(int a, int b, int c)

{

printf("Умножим обе части уравнения на а(%d), где ax=y\nПо теореме Виета:\n",a);

printf("\ty1\*y2 = a\*c = %d\n",a\*c);

printf("\ty1+y2 = -b = %d\n",-b);

int y1=0,y2=0;

if (a\*c>0)

{

while(y1\*y2!=a\*c || y1+y2!=-b)

{

b>0?y1--:y1++;

y2=(-b)-y1;

if (abs(y1)>10000)

{

printf("ОШИБКА! Корни не найдены\n");

return;

}

}

}

else

{

while(y1\*y2!=a\*c || y1+y2!=-b)

{

b>0?y1++:y1--;

y2=(-b)-y1;

if (abs(y1)>10000)

{

printf("ОШИБКА! Корни не найдены\n");

return;

}

}

}

printf("Находим y1 и y2: y1=%d; y2=%d\n",y1,y2);

printf("Получим корни: ");

if (y1==y2) printf("x=y/a=%.1f",(float)y1/a);

else printf("x1=y1/a=%.1f; x2=y2/a=%.1f\n",(float)y1/a,(float)y2/a);

}

void index\_props(int a, int b, int c)

{

printf("\tВНИМАНИЕ!\nМогут быть ошибки в расчетах c плавающей точкой!\n");

printf("Рассмотрим три свойства коэффициентов:\n");

if (a+b+c==0) printf("1). a+b+c=0 => x1=1; x2=c/a=%.1f\n",(float)c/a);

else printf("1). a+b+c!=0\n");

if (b%2==0)

{

int k = b/2;

printf("2). b = %d = 2k = 2\*%d - четное число, следовательно: x1=%.1f; x2=%.1f\n",b,k,(-k+sqrt(pow(k,2)-a\*c))/a,(-k-sqrt(pow(k,2)-a\*c)));

}

else printf("2). b = %d - нечетное число\n",b);

float p=b, q=c;

if(a!=1)

{

printf("3). Приведем уравнение\n");

p/=a;

q/=a;

}

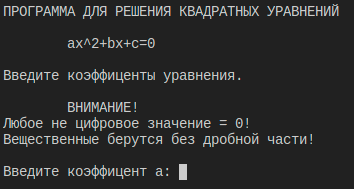
else printf("3). Это приведенное уравнение, где a=1, b=p, c=q\n");

printf("Следовательно, x1=%.1f, x2=%.1f\n",(-p+sqrt(pow(p,2)-4\*q))/2,(-p-sqrt(pow(p,2)-4\*q))/2);

}

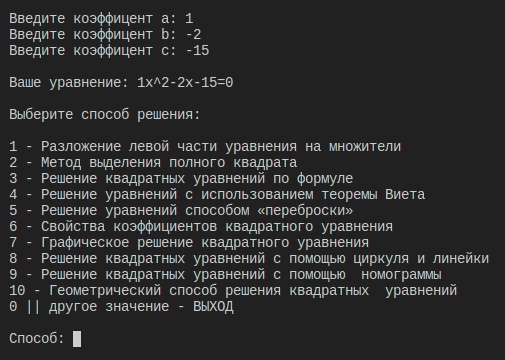
**Тестирование:**

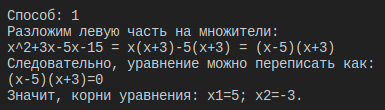
Программа открывается запуском исполняемого файла (рис. 1):

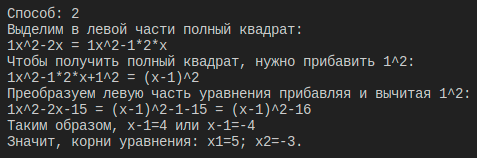


*Рис. 1. Запуск программы.*

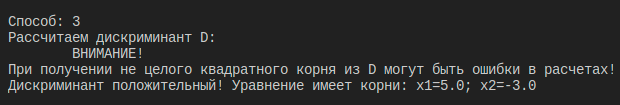
Сначала было протестировано решение уравнения с двумя корнями:  
x2 - 2x -15 = 0 (рис. 2-8):

*  
Рис. 2. Ввод коэффициентов.*

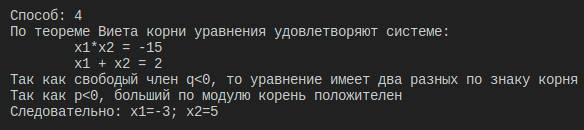
  
*Рис. 3. Разложение левой части уравнения на множители.*



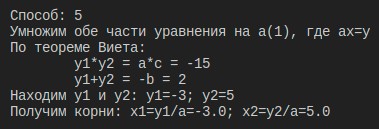
*Рис. 4. Метод выделения полного квадрата.*



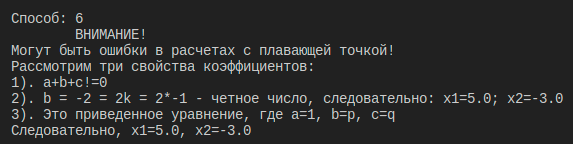
*Рис. 5. Решение квадратных уравнений по формуле.*



*Рис. 6. Решение уравнений с использованием теоремы Виета.*



*Рис. 7. Решение уравнений способом «переброски».*



*Рис. 8. Свойства коэффициентов квадратного уравнения.*

Графические способы (7-10) выводят сообщение о том, что они находятся на стадии разработки (рис. 9):

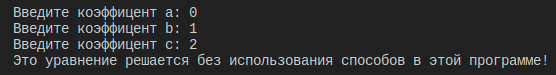


*Рис. 9. Графические способы решения.*

Выйти из программы можно с помощью команды 0 либо любого нецифрового ввода (рис. 10-11):

|  |  |
| --- | --- |
| *Рис. 10. Выход командой 0* | *Рис. 11. Выход текстовым вводом* |

При вводе какого-либо нулевого коэффициента программа сообщает пользователю об отсутствии необходимости приложения для данного уравнения (рис. 12):



*Рис. 12. Использование нулевых коэффициентов.*

**Вывод:** в ходе лабораторной работы мы научились создавать программы на языке Си для решения квадратных уравнений с помощью десяти различных методов, составлять блок-схему алгоритма программы и писать техническое задание по ГОСТ 19, а также составлять отчет по выполненной работе.