**Белорусский государственный технологический университет**

**Факультет информационных технологий**

**Кафедра программной инженерии**

**Программа “** Визуализирование результатов вычислений**”**

По дисциплине «Основы алгоритмизации и программирования»

Выполнил:

Студент 1 курса 10 группы ПИ

Макаревич Кирилл Витальевич

Проверил: Белодед Николай Иванович

Оглавление

[**Введение:** 3](#_Toc183730514)

[**Результат выполнения программы:** 4](#_Toc183730515)

[**Словесно-формульное описание алгоритма:** 5](#_Toc183730516)

[**Псевдоалгоритм:** 6](#_Toc183730517)

[**Блок-схема:** 7](#_Toc183730518)

[**Заключение:** 8](#_Toc183730519)

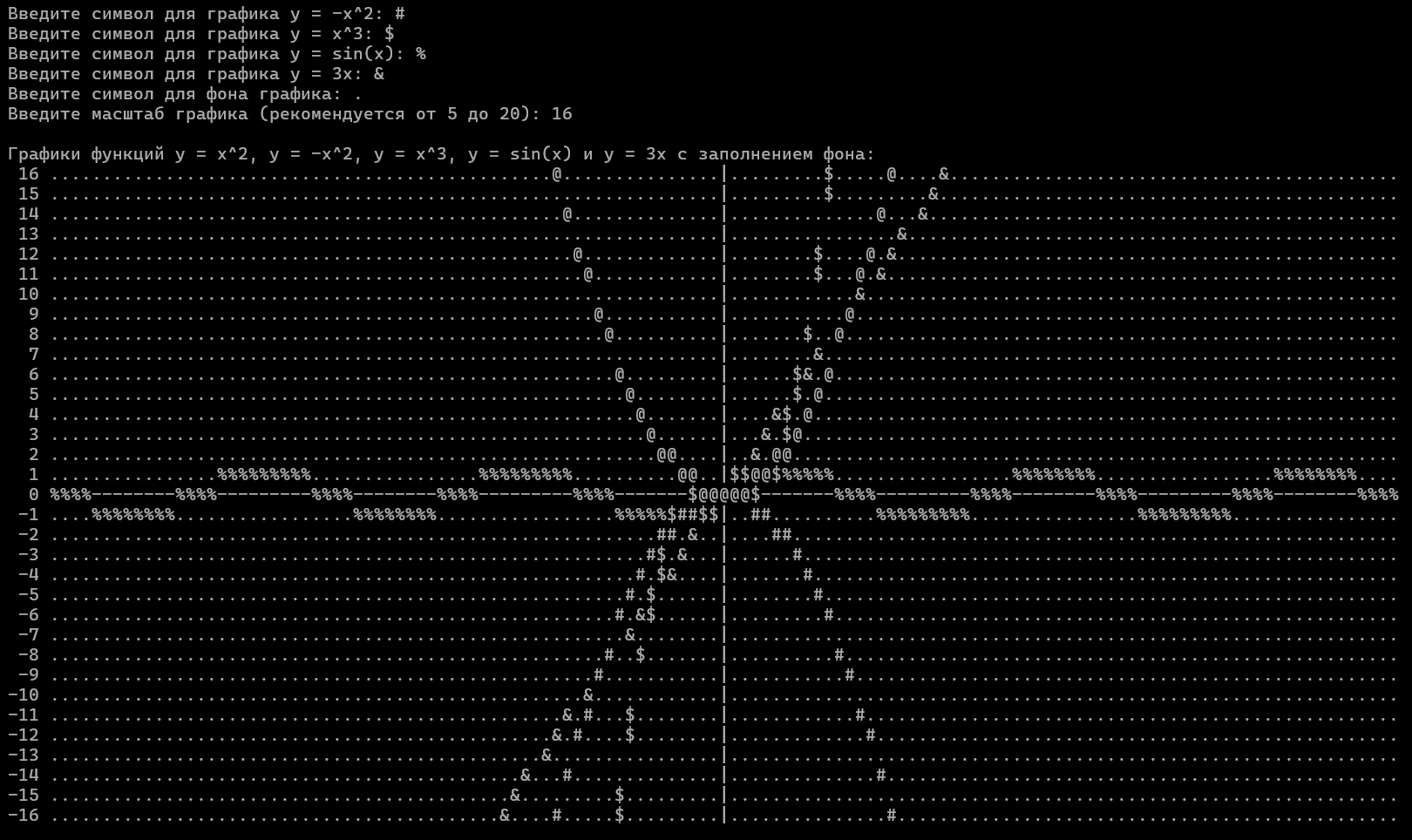
# **Введение:**

Целью данной работы является разработка программы на языке C++, которая визуализирует графики нескольких математических функций на координатной плоскости с заданными параметрами. Программа предоставляет пользователю возможность выбрать символы для отображения графиков функций, осей координат и фона, а также задавать масштаб. Такой подход помогает лучше понять поведение функций и их взаимное расположение на плоскости, что важно для изучения математики и программирования.

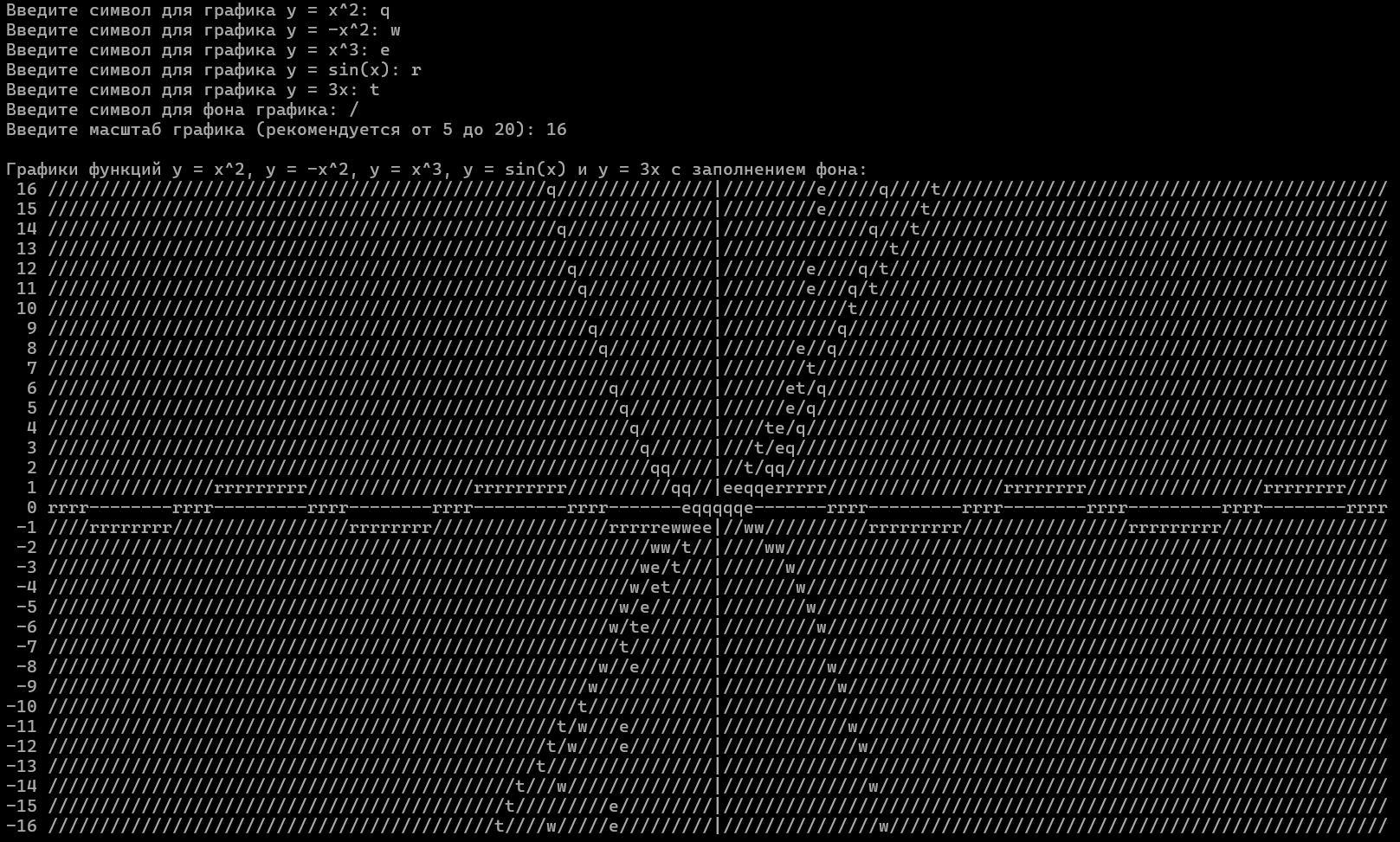
**Код программы:**

|  |
| --- |
| #include <iostream>  #include <cmath>  #include <iomanip>  using namespace std;  void drawGraph(char symbol1, char symbol2, char symbol3, char symbol4, char symbol5, char background, int scale) {  const int range = scale; // диапазон осей от -scale до scale  const double stepX = 0.25; // шаг для X  const double stepY = 1; // шаг для Y  // цикл по оси Y, начиная с верхней границы  for (double y = range; y >= -range; y -= stepY) {  // вывод метки оси Y  if (abs(y) < 0.1) {  cout << " 0 "; // центр оси  }  else {  cout << setw(3) << static\_cast<int>(round(y)) << " ";  }  // Цикл по оси X  for (double x = -range; x <= range; x += stepX) {  // Значение функций  double function1 = x \* x;  double function2 = -x \* x;  double function3 = x \* x \* x;  double function4 = sin(x);  double function5 = 3 \* x;  // отрисовка точки графика y = x^2  if (abs(function1 - y) < stepY / 2) {  cout << symbol1;  }  // отрисовка точки графика y = -x^2  else if (abs(function2 - y) < stepY / 2) {  cout << symbol2;  }  // отрисовка точки графика y = x^3  else if (abs(function3 - y) < stepY) {  cout << symbol3;  }  // отрисовка точки графика y = sin(x)  else if (abs(function4 - y) < stepY / 2) {  cout << symbol4;  }  // отрисовка точки графика y = 3x  else if (abs(function5 - y) < stepY / 2) {  cout << symbol5;  }  // отрисовка оси Y  else if (abs(x) < stepX / 2 && abs(y) > stepY / 2) {  cout << "|";  }  // отрисовка оси X  else if (abs(y) < stepY / 2 && abs(x) > stepX / 2) {  cout << "-";  }  // пересечение осей  else if (abs(x) < stepX / 2 && abs(y) < stepY / 2) {  cout << "+";  }  // пустое пространство заполняется фоном  else {  cout << background;  }  }  cout << "\n";  }  cout << "\n";  }  int main() {  setlocale(LC\_ALL, "ru");  char symbol1, symbol2, symbol3, symbol4, symbol5, background;  int scale;  cout << "Введите символ для графика y = x^2: ";  cin >> symbol1;  cout << "Введите символ для графика y = -x^2: ";  cin >> symbol2;  cout << "Введите символ для графика y = x^3: ";  cin >> symbol3;  cout << "Введите символ для графика y = sin(x): ";  cin >> symbol4;  cout << "Введите символ для графика y = 3x: ";  cin >> symbol5;  cout << "Введите символ для фона графика: ";  cin >> background;  cout << "Введите масштаб графика (рекомендуется от 5 до 20): ";  cin >> scale;  cout << "\nГрафики функций y = x^2, y = -x^2, y = x^3, y = sin(x) и y = 3x с заполнением фона:\n";  drawGraph(symbol1, symbol2, symbol3, symbol4, symbol5, background, scale);  return 0;  } |

# **Результат выполнения программы:**







# **Словесно-формульное описание алгоритма:**

1. Ввод параметров от пользователя:
   * Символы для отображения графиков:
     + y=x^2, y=-x^2, y= x^3, y=sin(x), y=3x.
   * Символ для заполнения пустого пространства графика (фона).
   * Масштаб (scale) для осей от scale до scale.
2. Настроить шаги:
   * stepX=0.25: шаг изменения xx для плавного построения графика.
   * stepY=1: шаг изменения yy для корректного отображения осей.
3. Построение графика:
   * Для каждой строки (значения y) от верхней границы (scale) до нижней (−scale):
     + Если y близко к нулю, вывести метку центра (0).
     + Иначе вывести метку текущего значения y.
   * Для каждого столбца (значения x):
     + Если точка соответствует одной из функций, вывести символ этой функции.
     + Если точка лежит на оси X или Y, вывести соответствующие символы осей (∣| для оси Y, −- для оси X).
     + Если точка — пересечение осей, вывести символ центра (+).
     + В противном случае заполнить фон символом, указанным пользователем.
4. Переход на новую строку:  
   После завершения цикла x переходим на следующую строку графика.
5. Завершение программы:  
   После вывода всех строк графика программа завершает работу.

# **Псевдоалгоритм:**

**НАЧАЛО**  
    **УСТАНОВИТЬ** кодировку консоли на Windows-1251

**СОЗДАТЬ** переменные:  
        symbol1, symbol2, symbol3, symbol4, symbol5, background, scale

**ВЫВЕСТИ**: "Введите символ для графика y = x^2:"  
    **ВВЕСТИ**: symbol1

**ВЫВЕСТИ**: "Введите символ для графика y = -x^2:"  
    **ВВЕСТИ**: symbol2

**ВЫВЕСТИ**: "Введите символ для графика y = x^3:"  
    **ВВЕСТИ**: symbol3

**ВЫВЕСТИ**: "Введите символ для графика y = sin(x):"  
    **ВВЕСТИ**: symbol4

**ВЫВЕСТИ**: "Введите символ для графика y = 3x:"  
    **ВВЕСТИ**: symbol5

**ВЫВЕСТИ**: "Введите символ для фона графика:"  
    **ВВЕСТИ**: background

**ВЫВЕСТИ**: "Введите масштаб графика (рекомендуется от 5 до 20):"  
    **ВВЕСТИ**: scale

**СОЗДАТЬ** параметры:  
        range = scale  
        stepX = 0.25  
        stepY = 1

**ДЛЯ** y **ОТ** range **ДО** -range **С ШАГОМ** -stepY:  
        **ЕСЛИ** |y| < 0.1 **ТО**  
            **ВЫВЕСТИ**: " 0 "  
        **ИНАЧЕ**  
            **ВЫВЕСТИ**: округленное значение y

**ДЛЯ** x **ОТ** -range **ДО** range **С ШАГОМ** stepX:  
            **ВЫЧИСЛИТЬ** значения функций:  
            function1 = x2x^2  
            function2 =-x^2  
            function3 =x^3  
            function4 =sin(x)  
            function5 = 3x

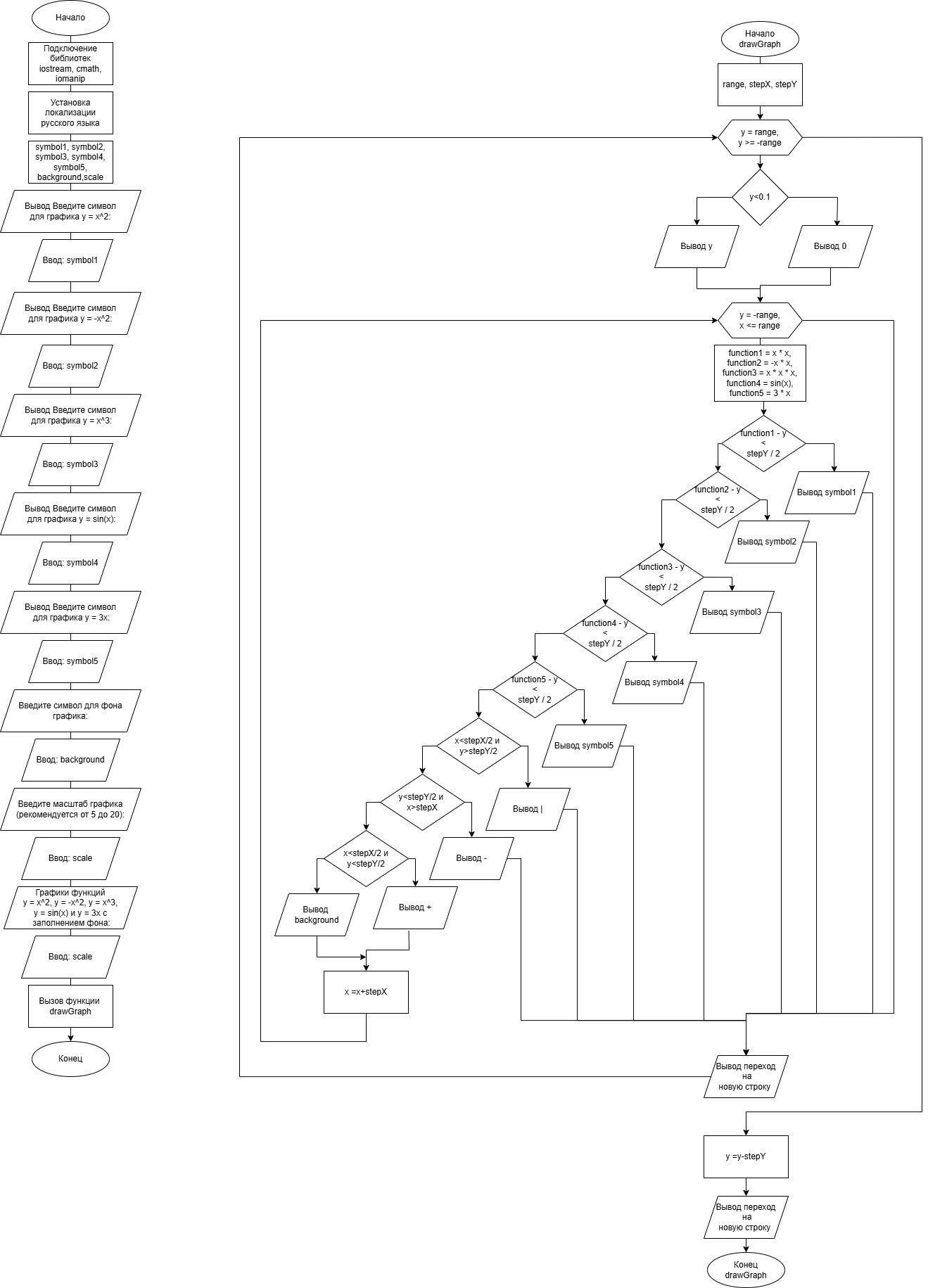
**ЕСЛИ** |function1 - y| < stepY / 2 **ТО**  
                **ВЫВЕСТИ**: symbol1  
            **ИНАЧЕ ЕСЛИ** |function2 - y| < stepY / 2 **ТО**  
                **ВЫВЕСТИ**: symbol2  
            **ИНАЧЕ ЕСЛИ** |function3 - y| < stepY **ТО**  
                **ВЫВЕСТИ**: symbol3  
            **ИНАЧЕ ЕСЛИ** |function4 - y| < stepY / 2 **ТО**  
                **ВЫВЕСТИ**: symbol4  
            **ИНАЧЕ ЕСЛИ** |function5 - y| < stepY / 2 **ТО**  
                **ВЫВЕСТИ**: symbol5  
            **ИНАЧЕ ЕСЛИ** |x| < stepX / 2 И |y| > stepY / 2 **ТО**  
                **ВЫВЕСТИ**: "|"  
            **ИНАЧЕ ЕСЛИ** |y| < stepY / 2 И |x| > stepX / 2 **ТО**  
                **ВЫВЕСТИ**: "-"  
            **ИНАЧЕ ЕСЛИ** |x| < stepX / 2 И |y| < stepY / 2 **ТО**  
                **ВЫВЕСТИ**: "+"  
            **ИНАЧЕ**  
                **ВЫВЕСТИ**: background

**ПЕРЕЙТИ** на следующую строку

**ВЫВЕСТИ** график

**КОНЕЦ**

# **Блок-схема:**



# **Заключение:**

Разработанная программа успешно выполняет поставленную задачу: визуализирует графики функций на координатной плоскости. Пользователь может настроить внешний вид графиков и масштаб. Применённый алгоритм продемонстрировал эффективность и гибкость, а процесс его реализации способствовал углублению знаний в области программирования и работы с графическим выводом в консоли.