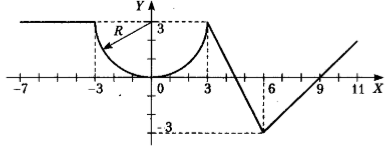
СОДЕРЖАНИЕ

**Лабораторная работа №2**

**Тема: «Разветвляющиеся вычислительные процессы »**

**Цель работы:** Изучение операторов ветвления на языке C#.

**Задание:**  Часть 1. Написать программу, вычисляющую значение кусочной функции, заданной на графике (Рисунок 1).

  
Рисунок 1 – График кусочной функции

Часть 2. Написать программу, которая определяет, принадлежит ли пара (х,у) закрашенной области (Рисунок 2).

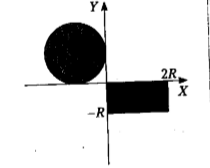


Рисунок 2 – Изображение закрашенной области

**Краткая теория**

Для организации условного ветвления язык C# унаследовал от С и С++ конструкцию if...else. Синтаксис:

if (условие)

оператор (операторы)

else

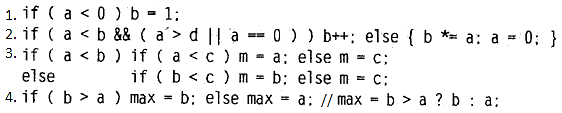
оператор (операторы)

Если по каждому из условий нужно выполнить более одного оператора, эти операторы должны быть объединены в блок с помощью фигурных скобок {...}.

Стоит обратить внимание, что в отличие от языков С и С++, в C# условный оператор if может работать только с булевскими выражениями, но не с произвольными значениями вроде -1 и 0.

В операторе if могут применяться сложные выражения, и он может содержать операторы else, обеспечивая выполнение более сложных проверок. Синтаксис похож на применяемый в аналогичных ситуациях в языках С (С++) и Java. При построении сложных выражений в C# используется вполне ожидаемый набор логических операторов.

Примеры использования условного оператора if:



Вторым оператором выбора в C# является оператор switch, который обеспечивает многонаправленное ветвление программы. Следовательно, этот оператор позволяет сделать выбор среди нескольких альтернативных вариантов дальнейшего выполнения программы. Несмотря на то, что многонаправленная проверка может быть организована с помощью последовательного ряда вложенных операторов if, во многих случаях более эффективным оказывается применение оператора switch. Этот оператор действует следующим образом. Значение выражения последовательно сравнивается с константами выбора из заданного списка. Как только будет обнаружено совпадение с одним из условий выбора, выполняется последовательность операторов, связанных с этим условием. Ниже приведена общая форма оператора switch:

switch(выражение) {

case константа1:

последовательность операторов

break;

case константа2:

последовательность операторов

break;

case константаЗ:

последовательность операторов

break;

default: //Введено что-то другое

последовательность операторов

break;

}

Оператор switch...case в C# он немного безопаснее, чем его эквивалент С++. В частности, он запрещает "сквозные" условия почти во всех случаях. Это значит, что если часть case вызывается в начале блока, то фрагменты кода за последующими частями case не могут быть выполнены, если только не используется явно оператор goto для перехода к ним. Компилятор обеспечивает это ограничение за счет того, что требует, чтобы за каждой частью case следовал оператор break, в противном случае он выдает ошибку.

Важно отметить, что заданное выражение в операторе switch должно быть целочисленного типа (char, byte, short или int), перечислимого или же строкового. А выражения других типов, например с плавающей точкой, в операторе switch не допускаются. Зачастую выражение, управляющее оператором switch, просто сводится к одной переменной. Кроме того, константы выбора должны иметь тип, совместимый с типом выражения. В одном операторе switch не допускается наличие двух одинаковых по значению констант выбора.

**Практическая часть**

Код программы:

using System;

namespace The\_saga

{

class Program

{

public static void Main(string[] args)

{

double x, y=0,R;

Console.WriteLine("Первая часть задания");

do

{

Console.Write("Введите x=");

x = Convert.ToDouble(Console.ReadLine());

if (x < -3)

y = 3;

else if (-3 <= x && x <= 3)

y = 3 - Math.Sqrt(9 - Math.Pow((x ), 2));

else if (3 < x && x < 6)

y = -x+1;

else if (6 <= x)

y = x+1;

Console.WriteLine("y=" +y);

Console.WriteLine("Приступить ко второй части задания? (Да - любой символ, Нет - нет)");

}

while (Console.ReadLine() == "нет");

Console.WriteLine("Вторая часть задания");

do

{

Console.Write("Введите R=");

R = Convert.ToDouble(Console.ReadLine());

Console.Write("Введите x=");

x = Convert.ToDouble(Console.ReadLine());

Console.Write("Введите у=");

y = Convert.ToDouble(Console.ReadLine());

if ((y <= Math.Pow((x+R)+(y-R),2)||(y<=2\*R\*(-R))))

Console.WriteLine("Пара (х;у) принадлежит области");

else

Console.WriteLine("Пара (х;у) не принадлежит области");

Console.WriteLine("Завершить задание? (Да - любой символ, Нет - нет)");

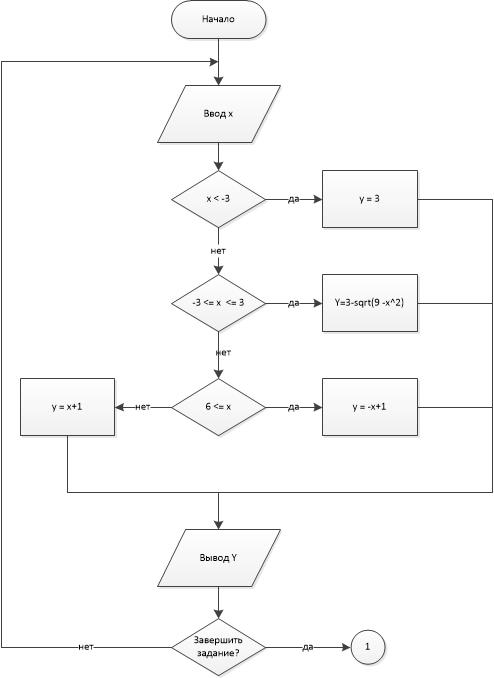
}

while (Console.ReadLine() == "нет");

}

}

}



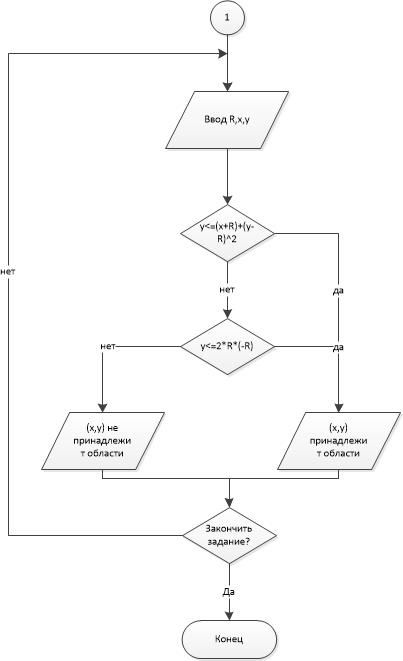


Рисунок 3 – Блок-схема реализации программы

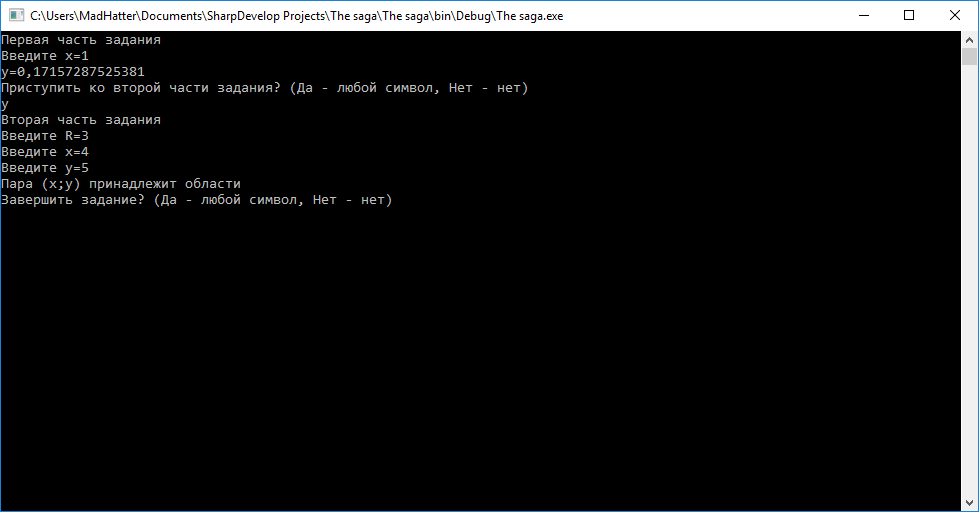


Рисунок 4 – Пример выполнения программы

**Вывод:** Получены навыки написания программ с ветвлениями на языке C# в среде программирования Microsoft Visual Studio.