**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**



**МОСКОВСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

**ВЫСШАЯ ШКОЛА ПЕЧАТИ И МЕДИАИНДУСТРИИ**

***Институт Принтмедиа и информационных технологий***

***Кафедра Информатики и информационных технологий***

**направление подготовки**

**09.03.02 «Информационные системы и технологии»**

**ЛАБОРАТОРНАЯ РАБОТА № 8**

**Дисциплина:** Введение в программирование.

**Выполнил(а):**

**студент(ка) группы 191-726**

Филатов. А.К.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  ***\_\_\_\_\_\_\_\_\_\_\_***

(Дата) (Подпись)

**Проверил:** асс. Кононенко К.М.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  ***\_\_\_\_\_\_\_\_\_\_\_***

(Дата) (Подпись)

**Замечания: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Москва**

**2019**

Оглавление

[**Теория** 3](#_Toc22550420)

[**Задания** 7](#_Toc22550431)

[**Блок-схемы** 8](#_Toc22550432)

[**Код программы** 12](#_Toc22550433)

[**Результат программы** 18](#_Toc22550434)

# 

# **Теория**

## **Логические операторы**

Следующие операторы выполняют логические операции с использованием логических операндов:

Унарный ! (логическое отрицание) оператор.

Бинарные & (логическое И), | (логическое ИЛИ), а также ^ (логическое исключающее ИЛИ) операторы. Эти операторы всегда обрабатывают оба операнда.

Бинарные && (условное логическое И) и || (условное логическое ИЛИ) операторы. Эти операторы вычисляют правый операнд, только если это необходимо.

Для операндов целочисленных типов операторы &, | и ^ выполняют побитовые логические операции.

## Оператор логического отрицания !

Унарный префиксный оператор ! выполняет логическое отрицание операнда, возвращая true, если операнд имеет значение false, и false, если операнд имеет значение true.

## Оператор логического И &

Оператор & вычисляет логическое И для всех своих операндов. Результат операции x & y принимает значение true, если оба оператора x и y имеют значение true. В противном случае результат будет false.

Оператор & вычисляет оба операнда, даже если левый операнд имеет значение false и результат должен принять значение false, независимо от значения правого операнда.

В следующем примере правый операнд оператора & является вызовом метода, который выполняется независимо от значения левого операнда:

Условный оператор логического И && также вычисляет логическое И для своих операндов, но не вычисляет правый операнд, если левый операнд имеет значение false.

Для операндов целочисленного типа оператор & вычисляет побитовое логическое И своих операндов. Унарный оператор & является оператором AddressOf.

## Оператор логического исключения ИЛИ ^

Оператор ^ вычисляет логическое исключение ИЛИ для всех своих операндов, возвращая true для x ^ y, если x имеет значение true и y имеет значение false или x имеет значение false и y имеет значение true. В противном случае результат будет false. То есть для операндов bool оператор ^ возвращает тот же результат, что и оператор неравенства !=.

Для операндов целочисленного типа оператор ^ вычисляет побитовое логическое исключающее ИЛИ своих операндов.

## Оператор логического ИЛИ |

Оператор | вычисляет логическое ИЛИ для всех своих операндов. Результат операции x | y принимает значение true, если хотя бы один из операторов x или y имеет значение true. В противном случае результат будет false.

Оператор | вычисляет оба операнда, даже если левый операнд имеет значение true и результат должен принять значение true, независимо от значения правого операнда.

В следующем примере правый операнд оператора | является вызовом метода, который выполняется независимо от значения левого операнда:

Условный оператор логического ИЛИ || также вычисляет логическое ИЛИ для своих операндов, но не вычисляет правый операнд, если левый операнд имеет значение true.

Для операндов целочисленного типа оператор | вычисляет побитовое логическое ИЛИ своих операндов.

## Условный оператор логического И &&

Условный оператор логического И && (оператор короткого замыкания) вычисляет логическое И для своих операндов. Результат операции x && y принимает значение true, если оба оператора x и y имеют значение true. В противном случае результат будет false. Если x имеет значение false, y не вычисляется.

В следующем примере правый операнд оператора && является вызовом метода, который не выполняется, если левый операнд имеет значение false:

Оператор логического И & также вычисляет логическое И для своих операндов, но он всегда вычисляет оба операнда.

## Условный оператор логического ИЛИ ||

Условный оператор логического ИЛИ || (оператор короткого замыкания) вычисляет логическое ИЛИ для своих операндов. Результат операции x || y принимает значение true, если хотя бы один из операторов x или y имеет значение true. В противном случае результат будет false. Если x имеет значение true, y не вычисляется.

В следующем примере правый операнд оператора || является вызовом метода, который не выполняется, если левый операнд имеет значение true:

Оператор логического ИЛИ | также вычисляет логическое ИЛИ для своих операндов, но всегда вычисляет оба операнда.

## Составное присваивание

Для бинарного оператора op выражение составного присваивания в форме за исключением того, что x вычисляется только один раз.

Операторы &, | и ^ поддерживают составное присваивание. Условные логические операторы && и || не поддерживают составное присваивание.

# **Задания**

1. Даны две переменные целого типа: A и B. Если их значения не равны, то присвоить каждой переменной большее из этих значений, а если равны, то присвоить переменным нулевые значения. Вывести новые значения переменных A и B.

2. Даны три числа. Найти сумму двух наибольших из них

3. На плоскости расположены три точки: A, B, C. Определить, какая из двух последних точек (B или C) расположена ближе к A, и вывести эту точку и ее расстояние от точки A.

4. Даны координаты точки, не лежащей на координатных осях OX и OY. Определить номер координатной четверти, в которой находится данная точка

5. Дано целое число. Вывести его строку-описание вида «отрицательное четное число», «нулевое число», «положительное нечетное число» и т. д.

6. Дано целое число, лежащее в диапазоне 1–999. Вывести его строку-описание вида «четное двузначное число», «нечетное трехзначное число» и т. д.

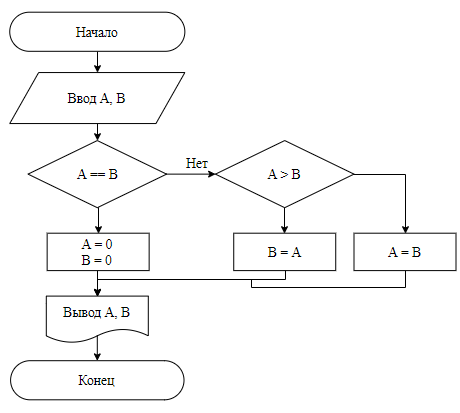
**Блок-схемы**  
 

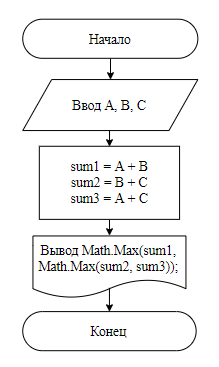
Рисунок 1 — Блок-схема к заданию 1  
 

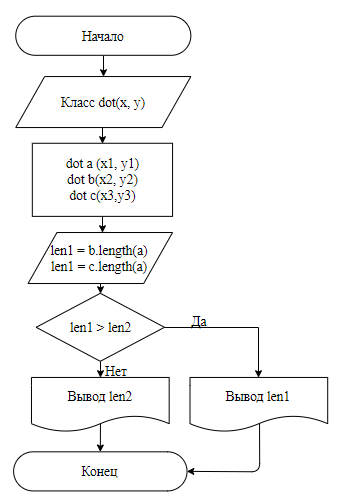
Рисунок 2 — Блок-схема к заданию 2   
 

Рисунок 3 — Блок-схема к заданию 3

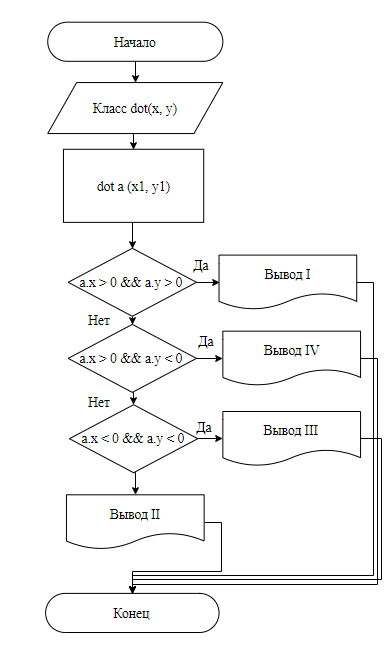


Рисунок 4 — Блок-схема к заданию 4

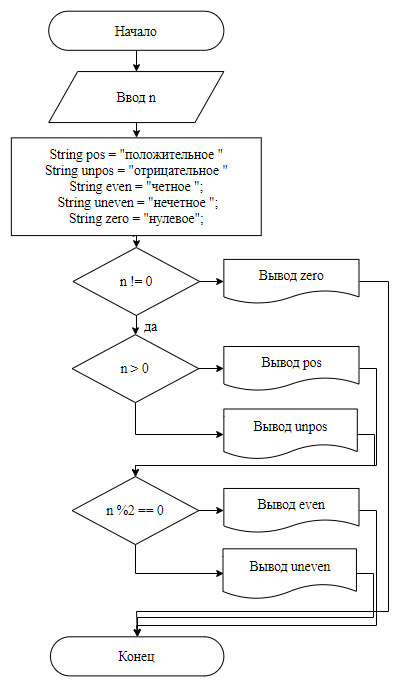


Рисунок 5 — Блок-схема к заданию 5

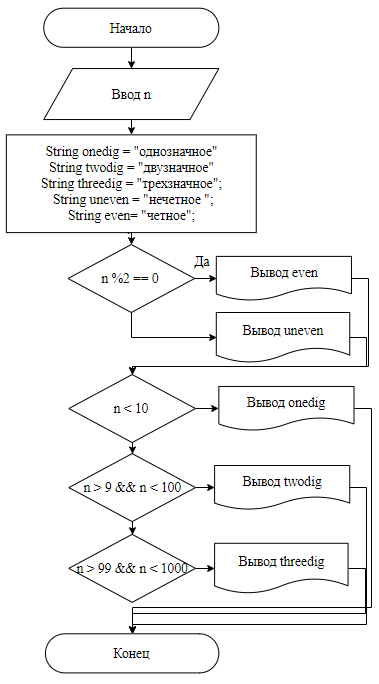


Рисунок 6 — Блок-схема к заданию 6

# **Код программы**

|  |
| --- |
| 1. using System; 2. using System.Collections.Generic; 3. using System.Linq; 4. using System.Text; 5. namespace LaboratorySolution 6. { 7. class Program 8. { 9. static void Main(string[] args) 10. { 11. int a, b; // создание переменных 12. a = int.Parse(Console.ReadLine()); 13. b = int.Parse(Console.ReadLine()); 14. if (a == b) 15. { 16. a = 0; 17. b = 0; 18. } 19. else 20. { 21. if (a > b) b = a; 22. else a = b; 23. } 24. Console.WriteLine(a + " " + b); 25. Console.ReadLine(); // пауза для просмотра результата 26. } 27. } 28. } |

Листинг 1 — Задание 1 (Вывести новые значения переменных A и B)

|  |
| --- |
| 1. using System; 2. using System.Collections.Generic; 3. using System.Linq; 4. using System.Text; 5. namespace LaboratorySolution 6. { 7. class Program 8. { 9. static void Main(string[] args) 10. { 11. int a, b, c, sum1, sum2, sum3; // создание переменных 12. a = int.Parse(Console.ReadLine()); 13. b = int.Parse(Console.ReadLine()); 14. c = int.Parse(Console.ReadLine()); 15. sum1 = a + b; 16. sum2 = b + c; 17. sum3 = a + c; 18. Console.WriteLine(Math.Max(sum1, Math.Max(sum2, sum3))); 19. Console.ReadLine(); // пауза для просмотра результата 20. } 21. } 22. } |

Листинг 2 — Задание 2 (Найти сумму двух наибольших чисел)

|  |
| --- |
| 1. using System; 2. using System.Collections.Generic; 3. using System.Linq; 4. using System.Text; 5. namespace LaboratorySolution 6. { 7. class Program 8. { 9. public class dot 10. { 11. public int x, y; 12. public dot() 13. { 14. x = 0; y = 0; 15. } 16. public double length(dot n) 17. { 18. return Math.Sqrt( Math.Pow(n.x - x, 2) + Math.Pow(n.y - y, 2)); 19. } 20. } 21. static void Main(string[] args) 22. { 23. dot a = new dot 24. { 25. x = int.Parse(Console.ReadLine()), 26. y = int.Parse(Console.ReadLine()) 27. }; 28. dot b = new dot 29. { 30. x = int.Parse(Console.ReadLine()), 31. y = int.Parse(Console.ReadLine()) 32. }; 33. dot c = new dot 34. { 35. x = int.Parse(Console.ReadLine()), 36. y = int.Parse(Console.ReadLine()) 37. }; 38. double len1 = b.length(a); 39. double len2 = c.length(a); 40. if (len1 < len2) Console.WriteLine("B, length = " + len1); 41. else Console.WriteLine("C, length = " + len2); 42. Console.ReadLine(); // пауза для просмотра результата 43. } 44. } 45. } |

Листинг 3 — Задание 3 (Вывести точку)

|  |
| --- |
| 1. using System; 2. using System.Collections.Generic; 3. using System.Linq; 4. using System.Text; 5. namespace LaboratorySolution 6. { 7. class Program 8. { 9. public class dot 10. { 11. public int x, y; 12. public dot() 13. { 14. x = 0; y = 0; 15. } 16. public double length(dot n) 17. { 18. return Math.Sqrt( Math.Pow(n.x - x, 2) + Math.Pow(n.y - y, 2)); 19. } 20. } 21. static void Main(string[] args) 22. { 23. dot a = new dot 24. { 25. x = int.Parse(Console.ReadLine()), 26. y = int.Parse(Console.ReadLine()) 27. }; 28. if (a.x > 0 && a.y > 0) Console.WriteLine("Quarter: I"); 29. else if (a.x > 0 && a.y < 0) Console.WriteLine("Quarter: IV"); 30. else if (a.x < 0 && a.y < 0) Console.WriteLine("Quarter: III"); 31. else if (a.x < 0 && a.y > 0) Console.WriteLine("Quarter: II"); 32. Console.ReadLine(); // пауза для просмотра результата 33. } 34. } 35. } |

Листинг 4 — Задание 4 (Определения номера координатной четверти, в которой находится введенная точка)

Листинг 5 — Задание 5 (Вывести его строку-описание)

|  |
| --- |
| 1. using System; 2. using System.Collections.Generic; 3. using System.Linq; 4. using System.Text; 5. namespace LaboratorySolution 6. { 7. class Program 8. { 9. static void Main(string[] args) 10. { 11. String pos = "положительное "; 12. String unpos = "отрицательное "; 13. String even = "четное "; 14. String uneven = "нечетное "; 15. String zero = "нулевое "; 16. int n; 17. n = int.Parse(Console.ReadLine()); 18. if (n != 0) 19. { 20. if (n > 0) Console.Write(pos); 21. else Console.Write(unpos); 22. if (n % 2 == 0) Console.Write(even); 23. else Console.Write(uneven); 24. } 25. else Console.Write(zero); 26. Console.Write("число"); 27. Console.ReadLine(); // пауза для просмотра результата 28. } 29. } 30. } |

Листинг 6 — Задание 6 (Вывести его строку-описание)

|  |
| --- |
| 1. using System; 2. using System; 3. using System.Collections.Generic; 4. using System.Linq; 5. using System.Text; 6. namespace LaboratorySolution 7. { 8. class Program 9. { 10. static void Main(string[] args) 11. { 12. String onedig = "однозначное "; 13. String twodig = "двузначное "; 14. String threedig = "трехзначное "; 15. String even = "четное "; 16. String uneven = "нечетное "; 17. int n = -1; 18. while (n < 1 || n > 999) 19. { 20. Console.WriteLine("Please input n in [1; 999] range!"); 21. n = int.Parse(Console.ReadLine()); 22. } 23. if (n % 2 == 0) Console.Write(even); 24. else Console.Write(uneven); 25. if (n < 10) Console.Write(onedig); 26. else if (n > 9 && n < 100) Console.Write(twodig); 27. else if (n > 99 && n < 1000) Console.Write(threedig); 28. Console.Write("число"); 29. Console.ReadLine(); // пауза для просмотра результата 30. } 31. } 32. } |

# **Результат программы**

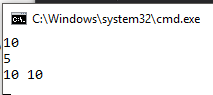


Рисунок 7 — Результат выполнения программы 1

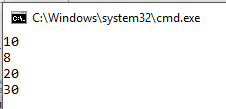


Рисунок 8 — Результат выполнения программы 2

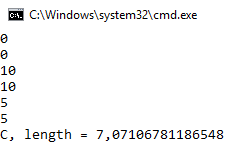


Рисунок 9 — Результат выполнения программы 3

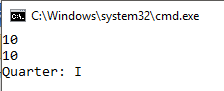


Рисунок 10 — Результат выполнения программы 4

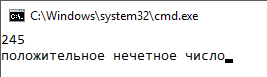


Рисунок 11 — Результат выполнения программы 5

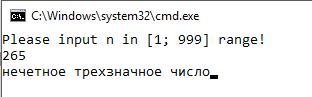


Рисунок 12 — Результат выполнения программы 6