**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**

Синельникова К.Т.

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

(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.

## Приоритет операторов

В следующем списке перечислены логические операторы в порядке убывания приоритета:

* Оператор логического отрицания !
* Оператор логического И &
* Оператор логического исключающего ИЛИ ^
* Оператор логического ИЛИ |
* Условный оператор логического И &&
* Условный оператор логического ИЛИ ||

Порядок вычисления, определяемый приоритетом операторов, можно изменить с помощью скобок (()).

# **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. using System.Threading.Tasks; 6. namespace Код\_Лабораторной\_8\_\_1\_6\_\_шарп 7. { 8. class Program 9. { 10. static void Main(string[] args) 11. { 12. int A, B; 13. Console.Write("Введите A: "); 14. A = int.Parse(Console.ReadLine()); 15. Console.Write("Введите B: "); 16. B = int.Parse(Console.ReadLine()); 17. if (A == B) 18. { 19. A = B = 0; 20. } 21. if (A > B) 22. { 23. B = A; 24. } 25. else 26. { 27. A = B; 28. } 29. Console.Write("Новые значения A и B: " + A + " " + B); 30. Console.ReadKey(); 31. } 32. } 33. } |

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. using System.Threading.Tasks; 6. namespace Код\_Лабораторной\_8\_\_1\_6\_\_шарп 7. { 8. class Program 9. { 10. static void Main(string[] args) 11. { 12. int A, B, C; 13. Console.Write("Введите первое число: "); 14. A = int.Parse(Console.ReadLine()); 15. Console.Write("Введите второе число: "); 16. B = int.Parse(Console.ReadLine()); 17. Console.Write("Введите третье число: "); 18. C = int.Parse(Console.ReadLine()); 19. if (A > C && B > C) 20. { 21. Console.WriteLine("Сумма наибольших чисел: " + (A + B)); 22. } 23. else if (B > A && C > A) 24. { 25. Console.WriteLine("Сумма наибольших чисел: " + (B + C)); 26. } 27. else 28. { 29. Console.WriteLine("Сумма наибольших чисел: " + (A + C)); 30. } 31. Console.ReadKey(); 32. } 33. } 34. } |

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. using System.Threading.Tasks; 6. namespace Код\_Лабораторной\_8\_\_1\_6\_\_шарп 7. { 8. class Program 9. { 10. static void Main(string[] args) 11. { 12. double AB, AC, x1, x2, x3, y1, y2, y3; 13. Console.WriteLine("Введите A (x1;y1): "); 14. x1 = int.Parse(Console.ReadLine()); 15. y1 = int.Parse(Console.ReadLine()); 16. Console.WriteLine("Введите B (x2;y2): "); 17. x2 = int.Parse(Console.ReadLine()); 18. y2 = int.Parse(Console.ReadLine()); 19. Console.WriteLine("Введите C (x3;y3): "); 20. x3 = int.Parse(Console.ReadLine()); 21. y3 = int.Parse(Console.ReadLine()); 22. AB = Math.Sqrt((x2 - x1) \* (x2 - x1) + (y2 - y1) \* (y2 - y1)); 23. AC = Math.Sqrt((x3 - x1) \* (x3 - x1) + (y3 - y1) \* (y3 - y1)); 24. if ( Math.Abs(AC) == Math.Abs(AB)) 25. { 26. Console.WriteLine("Точки равноудалены"); 27. } 28. else if (AB < AC) 29. { 30. Console.WriteLine("Ближайшая точка B. Расстояние от B до A = " + AB); 31. } 32. else 33. { 34. Console.WriteLine("Ближайшая точка C. Расстояние от C до A = " + AC); 35. } 36. Console.ReadKey(); 37. } 38. } 39. } |

Листинг 3 — Задание 3 (Вывести точку)

|  |
| --- |
| 1. using System; 2. using System.Collections.Generic; 3. using System.Linq; 4. using System.Text; 5. using System.Threading.Tasks; 6. namespace Код\_Лабораторной\_8\_\_1\_6\_\_шарп 7. { 8. class Program 9. { 10. static void Main(string[] args) 11. { 12. float x, y; 13. Console.WriteLine("Введите координаты точки, не лежащей на осях: "); 14. Console.Write("x = "); 15. x = float.Parse(Console.ReadLine()); 16. Console.Write("y = "); 17. y = float.Parse(Console.ReadLine()); 18. if (x>0) 19. { 20. if (y > 0) 21. { 22. Console.WriteLine("Точка лежит в первой четверти координатной системы"); 23. } 24. else 25. { 26. Console.WriteLine("Точка лежит в четвертой четверти координатной системы"); 27. } 28. } 29. else 30. { 31. if (y > 0) 32. { 33. Console.WriteLine("Точка лежит во второй четверти координатной системы"); 34. } 35. else 36. { 37. Console.WriteLine("Точка лежит в третьей четверти координатной системы"); 38. } 39. } 40. Console.ReadKey(); 41. } 42. } 43. } |

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. using System.Threading.Tasks; 6. namespace Код\_Лабораторной\_8\_\_1\_6\_\_шарп 7. { 8. class Program 9. { 10. static void Main(string[] args) 11. { 12. int A; 13. Console.Write("Введите A: "); 14. A = int.Parse(Console.ReadLine()); 15. if (A > 0) 16. { 17. if (A % 2 == 1) 18. { 19. Console.Write("Число " + A + " положительное нечетное"); 20. } 21. else 22. { 23. Console.Write("Число " + A + " положительное четное"); 24. } 25. } 26. else if (A < 0) 27. { 28. if (A % 2 == -1) 29. { 30. Console.Write("Число " + A + " отрицательное нечетное"); 31. } 32. else 33. { 34. Console.Write("Число " + A + " отрицательное четное"); 35. } 36. } 37. else Console.Write("Число A нулевое"); 38. Console.ReadKey(); 39. } 40. } 41. } |

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

|  |
| --- |
| 1. using System; 2. using System.Collections.Generic; 3. using System.Linq; 4. using System.Text; 5. using System.Threading.Tasks; 6. namespace Код\_Лабораторной\_7\_\_1\_7\_\_шарп 7. { 8. class Program 9. { 10. static void Main(string[] args) 11. { 12. int A; 13. Console.Write("Введите число A в диапазоне 1-999: "); 14. A = int.Parse(Console.ReadLine()); 15. if (A % 2 == 0) 16. { 17. if (A >= 100 && A < 1000) 18. { 19. Console.Write("Число " + A + " четное трехзначное"); 20. } 21. else if (A >= 10 && A < 100) 22. { 23. Console.Write("Число " + A + " четное двузначное"); 24. } 25. else 26. { 27. Console.Write("Число " + A + " четное однозначное"); 28. } 29. } 30. else 31. { 32. if (A >= 100 && A < 1000) 33. { 34. Console.Write("Число " + A + " нечетное трехзначное"); 35. } 36. else if (A >= 10 && A < 100) 37. { 38. Console.Write("Число " + A + " нечетное двузначное"); 39. } 40. else 41. { 42. Console.Write("Число " + A + " нечетное однозначное"); 43. } 44. } 45. Console.ReadKey(); 46. } 47. } 48. } |

# **Результат программы**

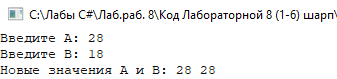


Рисунок 7 — Результат выполнения программы 1

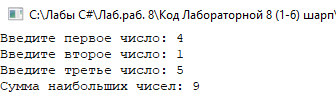


Рисунок 8 — Результат выполнения программы 2

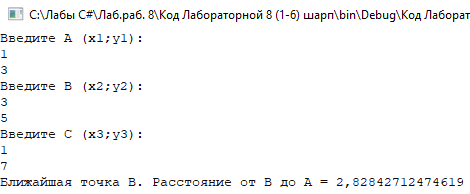


Рисунок 9 — Результат выполнения программы 3

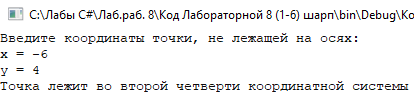


Рисунок 10 — Результат выполнения программы 4

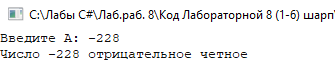


Рисунок 11 — Результат выполнения программы 5



Рисунок 12 — Результат выполнения программы 6