**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\_\_**

\_\_\_\_\_Мухина Н.А.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(Фамилия И.О.)

**Дата, подпись** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  ***\_\_\_\_\_\_\_\_\_\_\_***

(Дата) (Подпись)

**Проверил: \_\_*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_***

(Фамилия И.О., степень, звание) **(Оценка)**

**Дата, подпись** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  ***\_\_\_\_\_\_\_\_\_\_\_***

(Дата) (Подпись)

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

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

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

**Москва**

**2019**

Оглавление

[Теория: 3](#_Toc22814440)

[Задания: 3](#_Toc22814441)

[Блок-схемы: 4](#_Toc22814442)

[Код программы: 5](#_Toc22814443)

[Результат программы: 12](#_Toc22814449)

# Теория:

Операторы инкремента (++) и декремента (--) были представлены в главе 2. Как станет ясно в дальнейшем, они обладают рядом особых и довольно интересных свойств. Но сначала выясним основное назначение этих операторов. Оператор инкремента увеличивает свой операнд на 1, а оператор декремента уменьшает операнд на 1. Следовательно, оператор х++;

Глава 4. Операторы 99

равнозначен оператору х = x + 1; а оператор х--; равносилен оператору х = х - 1; Следует, однако, иметь в виду, что в инкрементной или декрементной форме значение переменной х вычисляется только один, а не два раза. В некоторых случаях это позволяет повысить эффективность выполнения программы. Оба оператора инкремента и декремента можно указывать до операнда (в префиксной форме) или же после операнда (в постфиксной форме). Например, оператор х = х + 1; может быть записан в следующем виде: ++х; // префиксная форма или же в таком виде: х++; // постфиксная форма В приведенном выше примере форма инкремента (префиксная или постфиксная) особого значения не имеет. Но если оператор инкремента или декремента используется в длинном выражении, то отличие в форме его записи уже имеет значение. Когда оператор инкремента или декремента предшествует своему операнду, то результатом операции становится значение операнда после инкремента или декремента. А когда оператор инкремента или декремента следует после своего операнда, то результатом операции становится значение операнда до инкремента или декремента. Рассмотрим следующий фрагмент кода. х = 10; у = ++х; В данном случае значение переменной у будет установлено равным 11, поскольку значение переменной х сначала увеличивается на 1, а затем присваивается переменной у. Но во фрагменте кода х = 10; у = х++; значение переменной у будет установлено равным 10, так как в этом случае значение переменной х сначала присваивается переменной у, а затем увеличивается на 1. В обоих случаях значение переменной х оказывается равным 11. Отличие состоит лишь том, когда именно это значение станет равным 11: до или после его присваивания переменной у. Возможность управлять моментом инкремента или декремента дает немало преимуществ при программировании. Обратимся к следующему примеру программы, в которой формируется последовательный ряд чисел. // Продемонстрировать отличие между префиксной // и постфиксной формами оператора инкремента (++).

В C# имеются две общие категории встроенных типов данных: типы значений и ссылочные типы. Они отличаются по содержимому переменной. Если переменная относится к типу значения, то она содержит само значение, например, 3,1416 или 212. А если переменная относится к ссылочному типу, то она содержит ссылку на значение. Наиболее распространенным примером использования ссылочного типа является класс, но о классах и ссылочных типах речь пойдет далее в этой книге. А здесь рассматриваются типы значений. В основу языка C# положены 13 типов значений. Все они называются простыми типами, поскольку состоят из единственного значения. (Иными словами, они не состоят из двух или более значений.) Они составляют основу системы типов С#, предоставляя простейшие, низкоуровневые элементы данных, которыми можно оперировать в программе. Простые типы данных иногда еще называют примитивными.

В C# строго определены пределы и характер действия каждого типа значения. Исходя из требований к переносимости программ, C# не допускает в этом отношении никаких компромиссов. Например, тип int должен быть одинаковым во всех средах выполнения. Но в этом случае отпадает необходимость переписывать код для конкретной платформы. И хотя строгое определение размерности типов значений может стать причиной незначительного падения производительности в некоторых средах, эта мера необходима для достижения переносимости программ.

Помимо простых типов, в C# определены еще три категории типов значений: перечисления, структуры и обнуляемые типы.

Целочисленные типы со знаком имеют большое значение для очень многих алгоритмов, но по абсолютной величине они наполовину меньше своих аналогов без знака. Вот как, например, выглядит число 32 767 типа short в двоичном представлении. 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 Если установить старший разряд этого числа равным 1, чтобы получить значение со знаком, то оно будет интерпретировано как -1, принимая во внимание формат дополнения до двух. Но если объявить его как значение типа ushort, то после установки в 1 старшего разряда оно станет равным 65 535.

# 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:



Рис.1-Блок-схема



Рис.2-Блок-схема к заданиям 2,4,5,6



Рис.3-Блок-схема

Код программы:

Листинг 1 - Задание 1

|  |
| --- |
| 1. using System; 2. using System.Collections.Generic; 3. using System.Linq; 4. using System.Text; 5. using System.Threading.Tasks; 6. namespace \_1 7. { 8. class Program 9. { 10. static void Main(string[] args) 11. { 12. int A, B; 13. Console.WriteLine("Введите значения A и B : "); 14. A = int.Parse(Console.ReadLine()); 15. B = int.Parse(Console.ReadLine()); 16. if (A == B) 17. { 18. A = 0; 19. B = 0; 20. Console.WriteLine("A,B: {0}{1}", A, B); 21. } 22. else if(A>B) 23. { 24. B = A; 25. Console.WriteLine("A,B: {0}{1}", A, B); 26. } 27. else 28. { 29. A = B; 30. Console.WriteLine("A,B: {0}{1}", A, B); 31. } 32. Console.ReadKey(); 33. } 34. } 35. } |

# Листинг 2 – Задание 2

|  |
| --- |
| 1. using System; 2. using System.Collections.Generic; 3. using System.Linq; 4. using System.Text; 5. using System.Threading.Tasks; 6. namespace \_2 7. { 8. class Program 9. { 10. static void Main(string[] args) 11. { 12. int A, B, C, D; 13. Console.WriteLine("Введите значения A,B,C: "); 14. A = int.Parse(Console.ReadLine()); 15. B = int.Parse(Console.ReadLine()); 16. C = int.Parse(Console.ReadLine()); 17. D = 0; 18. if((A>B)&&(B>C)) 19. { 20. D = A + B; 21. Console.WriteLine("1Сумма наибольших чисел равна : " + D); 22. } 23. else if ((C > B) && (B > A)) 24. { 25. D = B + C; 26. Console.WriteLine("2Сумма наибольших чисел равна : " + D); 27. } 28. if ((B>A)&&(A>C)) 29. { 30. D = A + B; 31. Console.WriteLine("3Сумма наибольших чисел равна : " + D); 32. } 33. else if ((C > A) && (A > B)) 34. { 35. D = A + C; 36. Console.WriteLine("4Сумма наибольших чисел равна : " + D); 37. } 38. if ((A>C)&&(C>B)) 39. { 40. D = A + C; 41. Console.WriteLine("5Сумма наибольших чисел равна : " + D); 42. } 43. else if ((B > C) && (C > A)) 44. { 45. D = B + C; |

Продолжение Листинга 2

|  |
| --- |
| 1. Console.WriteLine("6Сумма наибольших чисел равна : " + D); 2. } 3. Console.ReadKey(); 4. }}} |

# Листинг 3 – Задание 3

|  |
| --- |
| 1. using System; 2. using System.Collections.Generic; 3. using System.Linq; 4. using System.Text; 5. using System.Threading.Tasks; 6. namespace \_3 7. { 8. class Program 9. { 10. static void Main(string[] args) 11. { 12. int A, B, C, AB, AC; 13. Console.WriteLine("Введите значения A,B,C: "); 14. A = int.Parse(Console.ReadLine()); 15. B = int.Parse(Console.ReadLine()); 16. C = int.Parse(Console.ReadLine()); 17. AB =Math.Abs ( B-A ); 18. AC =Math.Abs ( C-A ); 19. if(AB>AC) 20. { 21. Console.WriteLine("Точка C ближе к A : " + C); 22. Console.WriteLine("Расстояние равно : " + AC); 23. } 24. else 25. { 26. Console.WriteLine("Точка B ближе к A : " + B); 27. Console.WriteLine("Расстояние равно : " + AB); 28. } 29. Console.ReadKey(); 30. } 31. } 32. } |

# Листинг 4 – Задание 4

|  |
| --- |
| 1. using System; 2. using System.Collections.Generic; 3. using System.Linq; 4. using System.Text; 5. using System.Threading.Tasks; 6. namespace \_4 7. { 8. class Program 9. { 10. static void Main(string[] args) 11. { 12. int x, y; 13. Console.WriteLine("Введите координаты точки x и y: "); 14. x = int.Parse(Console.ReadLine()); 15. y = int.Parse(Console.ReadLine()); 16. if ((x > 0) && (y > 0)) 17. { 18. Console.WriteLine("Точка находится в 1 четверти "); 19. } 20. else if ((x < 0) && (y > 0)) 21. { 22. Console.WriteLine("Точка находится во 2 четверти"); 23. } 24. else if ((x < 0) && (y < 0)) 25. { 26. Console.WriteLine("Точка находится во 3 четверти"); 27. } 28. else if ((x > 0) && (y < 0)) 29. { 30. Console.WriteLine("Точка находится во 4 четверти"); 31. } 32. Console.ReadKey(); 33. } 34. } 35. } |

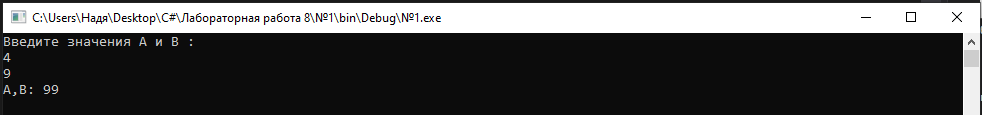
# Листинг 5 – Задание 5

|  |
| --- |
| 1. using System; 2. using System.Collections.Generic; 3. using System.Linq; 4. using System.Text; 5. using System.Threading.Tasks; 6. namespace \_5 7. { 8. class Program 9. { 10. static void Main(string[] args) 11. { 12. int x; 13. Console.WriteLine("Введите значение числа x: "); 14. x = int.Parse(Console.ReadLine()); 15. if(x==0) 16. { 17. Console.WriteLine("Нулевое число"); 18. } 19. if((x>0)&&(x%2==0)) 20. { 21. Console.WriteLine("Положительное четное число"); 22. } 23. if((x>0)&&(x%2!=0)) 24. { 25. Console.WriteLine("Положительное нечетное число"); 26. } 27. if ((x < 0) && (x % 2 == 0)) 28. { 29. Console.WriteLine("Отрицательное четное число"); 30. } 31. if ((x < 0) && (x % 2 != 0)) 32. { 33. Console.WriteLine("Отрицательное нечетное число"); 34. } 35. Console.ReadKey(); 36. } 37. } 38. } |

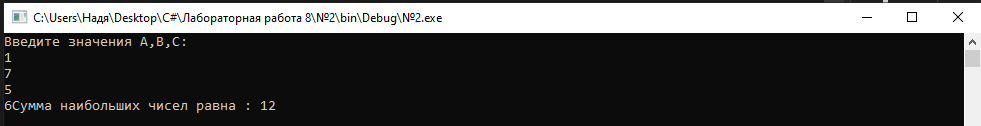
# Листинг 6 – Задание 6

|  |
| --- |
| 1. using System; 2. using System.Collections.Generic; 3. using System.Linq; 4. using System.Text; 5. using System.Threading.Tasks; 6. namespace \_6 7. { 8. class Program 9. { 10. static void Main(string[] args) 11. { 12. int x; 13. Console.WriteLine("Введите значение x: "); 14. x = int.Parse(Console.ReadLine()); 15. if((x>0)&&(x<10)&&(x%2==0)) 16. { 17. Console.WriteLine("Однозначное четное число"); 18. } 19. if ((x > 0) && (x < 10) && (x % 2 != 0)) 20. { 21. Console.WriteLine("Однозначное нечетное число"); 22. } 23. if ((x > 9) && (x < 100) && (x % 2 == 0)) 24. { 25. Console.WriteLine("Двузначное четное число"); 26. } 27. if ((x > 9) && (x < 100) && (x % 2 != 0)) 28. { 29. Console.WriteLine("Двузначное нечетное число"); 30. } 31. if ((x > 99) && (x < 1000) && (x % 2 == 0)) 32. { 33. Console.WriteLine("Трехзначное четное число"); 34. } 35. if ((x > 99) && (x < 1000) && (x % 2 != 0)) 36. { 37. Console.WriteLine("Трехзначное нечетное число"); 38. } 39. Console.ReadKey(); 40. } 41. } 42. } |

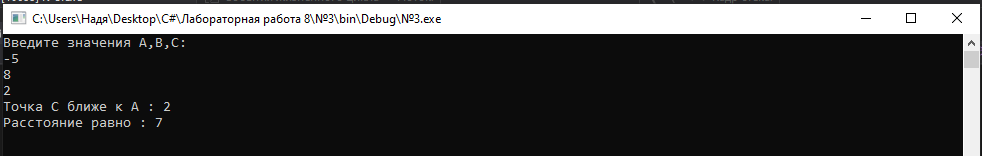
# Результат программы:



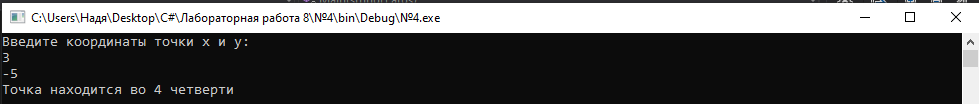
**Рис.4-Скриншот работы программы Задание №1**



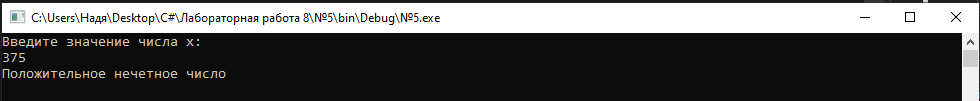
**Рис.5-Скриншот работы программы Задание №2**



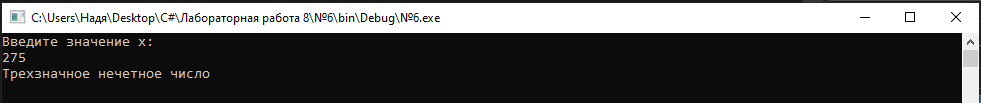
**Рис.6-Скриншот работы программы Задание №3**



**Рис.7-Скриншот работы программы Задание №4**



**Рис.8-Скриншот работы программы Задание №5**



**Рис.9-Скриншот работы программы Задание №6**