**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 № 7**

**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](#_Toc22550572)

[**Задания** 7](#_Toc22550583)

[**Блок-схемы** 8](#_Toc22550584)

[**Код программы** 12](#_Toc22550585)

[**Результат программы** 16](#_Toc22550586)

# 

# **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.

# **Задания**

1. Даны два целых числа: A, B. Проверить истинность высказывания: «Справедливы неравенства A > 2 и B ≤ 3»

2. Даны три целых числа: A, B, C. Проверить истинность высказывания: «Справедливо двойное неравенство A < B < C».

3. Дано целое положительное число. Проверить истинность высказывания: «Данное число является четным двузначным».

4. Дано трехзначное число. Проверить истинность высказывания: «Цифры данного числа образуют возрастающую или убывающую последовательность».

5. Дано четырехзначное число. Проверить истинность высказывания: «Данное число читается одинаково слева направо и справа налево».

6. Даны целые числа a, b, c, являющиеся сторонами некоторого треугольника. Проверить истинность высказывания: «Треугольник со сторонами a, b, c является прямоугольным».

7. Даны целые числа a, b, c. Проверить истинность высказывания: «Существует треугольник со сторонами a, b, c».

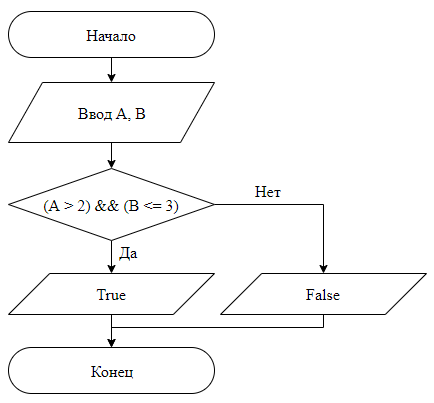
**Блок-схемы** 

Рисунок 1 — Блок-схема к заданию 1

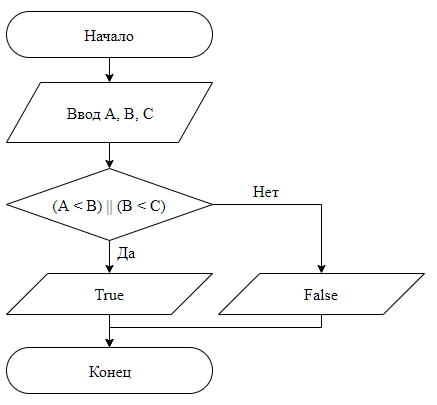


Рисунок 2 — Блок-схема к заданию 2

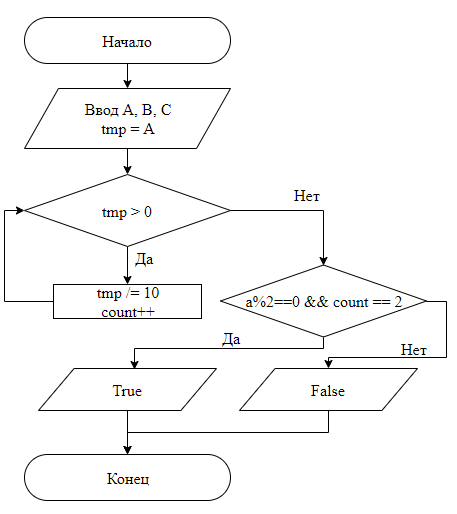


Рисунок 3 — Блок-схема к заданию 3

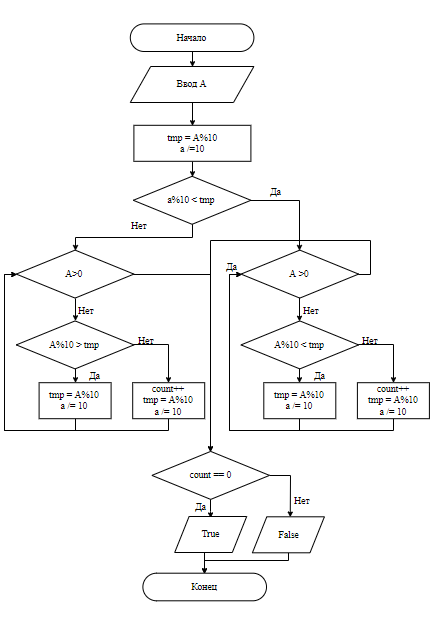


Рисунок 4 — Блок-схема к заданию 4

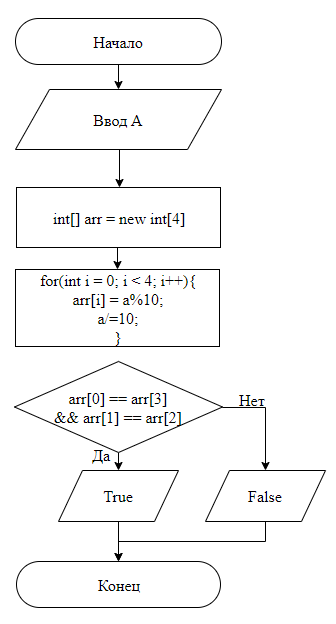


Рисунок 5 — Блок-схема к заданию 5

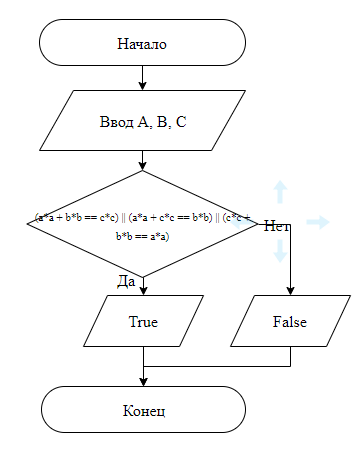


Рисунок 6 — Блок-схема к заданию 6

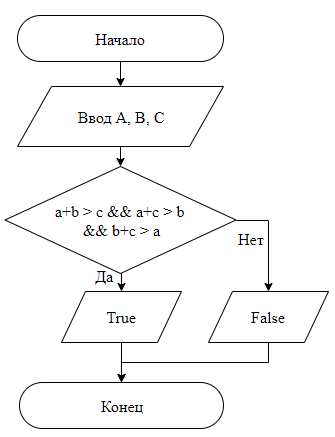


Рисунок 7 — Блок-схема к заданию 7

# **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 > 2 && b <= 3) Console.WriteLine("A > 2 and B <= 3 is TRUE"); 15. else Console.WriteLine("A > 2 and B <= 3 is FALSE"); 16. Console.ReadLine(); // пауза для просмотра результата 17. } 18. } 19. } |

Листинг 1 — Задание 1 (Справедливы неравенства A > 2 и B ≤ 3)

|  |
| --- |
| 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; // создание переменных 12. a = int.Parse(Console.ReadLine()); 13. b = int.Parse(Console.ReadLine()); 14. c = int.Parse(Console.ReadLine()); 15. if (a < b && b < c) Console.WriteLine("A < B < C is TRUE"); 16. else Console.WriteLine("A < B < C is FALSE"); 17. Console.ReadLine(); // пауза для просмотра результата 18. } 19. } 20. } |

Листинг 2 — Задание 2 (Справедливо двойное неравенство A < B < C)

|  |
| --- |
| 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, tmp, count = 0; // создание переменных 12. a = int.Parse(Console.ReadLine()); 13. tmp = a; 14. while(tmp > 0) 15. { 16. tmp /= 10; 17. count++; 18. } 19. if (a % 2 == 0 && count == 2) Console.WriteLine("Number is even and has two digits - TRUE"); 20. else Console.WriteLine("Number is even and has two digits - FALSE"); 21. Console.ReadLine(); // пауза для просмотра результата 22. } 23. } 24. } |

Листинг 3 — Задание 3 (Данное число является четным двузначным)

|  |
| --- |
| 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, tmp, count = 0; // создание переменных 12. a = int.Parse(Console.ReadLine()); 13. tmp = a % 10; 14. a /= 10; 15. if(a%10 < tmp) 16. { 17. while (a > 0) 18. { 19. if(a%10 < tmp) 20. { 21. tmp = a % 10; 22. a /= 10; 23. } 24. else 25. { 26. count++; 27. tmp = a % 10; 28. a /= 10; 29. } 30. } 31. if (count == 0) Console.WriteLine("TRUE"); 32. else Console.WriteLine("FALSE"); 33. } 34. else 35. { 36. while (a > 0) 37. { 38. if (a % 10 > tmp) 39. { 40. tmp = a % 10; 41. a /= 10; 42. } 43. else 44. { 45. count++; 46. tmp = a % 10; 47. a /= 10; 48. } 49. } 50. if (count == 0) Console.WriteLine("TRUE"); 51. else Console.WriteLine("FALSE"); 52. } 53. Console.ReadLine(); // пауза для просмотра результата 54. } 55. } 56. } |

Листинг 4 — Задание 4 (Цифры данного числа образуют возрастающую или убывающую последовательность)

Листинг 5 — Задание 5 (Данное число читается одинаково слева направо и справа налево)

|  |
| --- |
| 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; // создание переменных 12. a = int.Parse(Console.ReadLine()); 13. int[] arr = new int[4]; 14. for (int i = 0; i<4; i++) 15. { 16. arr[i] = a % 10; 17. a /= 10; 18. } 19. if (arr[0] == arr[3] && arr[1] == arr[2]) Console.WriteLine("TRUE"); 20. else Console.WriteLine("FALSE"); 21. Console.ReadLine(); // пауза для просмотра результата 22. } 23. } 24. } |

# 

Листинг 6 — Задание 6 (Треугольник со сторонами a, b, c является прямоугольным)

|  |
| --- |
| 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; // создание переменных 12. a = int.Parse(Console.ReadLine()); 13. b = int.Parse(Console.ReadLine()); 14. c = int.Parse(Console.ReadLine()); 15. if ((a\*a + b\*b == c\*c) || (a\*a + c\*c == b\*b) || (c\*c + b\*b == a\*a)) Console.WriteLine("TRUE"); 16. else Console.WriteLine("FALSE"); 17. Console.ReadLine(); // пауза для просмотра результата 18. } 19. } 20. } |

Листинг 7 — Задание 7 (Существует треугольник со сторонами a, b, c)

|  |
| --- |
| 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; // создание переменных 12. a = int.Parse(Console.ReadLine()); 13. b = int.Parse(Console.ReadLine()); 14. c = int.Parse(Console.ReadLine()); 15. if (a+b > c && a+c > b && b+c > a) Console.WriteLine("TRUE"); 16. else Console.WriteLine("FALSE"); 17. Console.ReadLine(); // пауза для просмотра результата 18. } 19. } 20. } |

# **Результат программы**

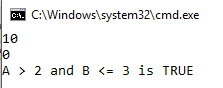


Рисунок 8 — Результат выполнения программы 1

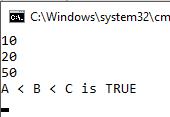


Рисунок 9 — Результат выполнения программы 2

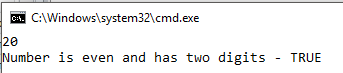


Рисунок 10 — Результат выполнения программы 3

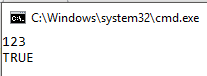


Рисунок 11 — Результат выполнения программы 4

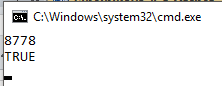


Рисунок 12 — Результат выполнения программы 5

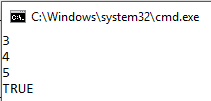


Рисунок 13 — Результат выполнения программы 6

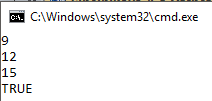


Рисунок 14 — Результат выполнения программы 7