**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 № 6**

**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** 2](#_Toc20500756)

[**Задания** 4](#_Toc20500757)

[**Блок-схемы** 5](#_Toc20500758)

[**Код программы** 9](#_Toc20500759)

[**Результат программы** 14](#_Toc20500760)

# 

# **Теория**

Составные операторы присваивания

В C# предусмотрены специальные составные операторы присваивания, упрощающие программирование некоторых операций присваивания.

Пара операторов += указывает компилятору на то, что переменной х должно быть присвоено ее первоначальное значение, увеличенное на 1.

Для многих двоичных операций, т.е. операций, требующих наличия двух операндов, существуют отдельные составные операторы присваивания. Общая форма всех этих операторов имеет следующий вид:

*имя переменной op = выражение*

где op — арифметический или логический оператор, применяемый вместе с оператором присваивания.

|  |  |
| --- | --- |
| **Оператор** | **Аналог (выражение из вышеуказанного примера)** |
| += | x = x + 1; |
| -= | x = x - 1; |
| \*= | x = x\*1; |
| /= | x = x/1; |
| %= | x = x%1; |
| |= | x = x | 1; |
| ^= | x = x^1; |

Составные операторы присваивания записываются более кратко, чем их несоставные эквиваленты. Поэтому их иногда еще называют укороченными операторами присваивания.

У составных операторов присваивания имеются два главных преимущества. Во-первых, они более компактны, чем их "несокращенные" эквиваленты. И во-вторых, они дают более эффективный исполняемый код, поскольку левый операнд этих операторов вычисляется только один раз. Именно по этим причинам составные операторы присваивания чаще всего применяются в программах, профессионально написанных на C#.

# **Задания**

1. Найти расстояние между двумя точками с заданными координатами (x1, y1) и (x2, y2)

2. Даны три точки A, B, C на числовой оси. Найти длины отрезков AC

и BC и их сумму.

3. Даны три точки A, B, C на числовой оси. Точка C расположена

между точками A и B. Найти произведение длин отрезков AC и BC

4. Даны координаты двух противоположных вершин прямоугольника:

(x1, y1), (x2, y2). Стороны прямоугольника параллельны осям координат.

Найти периметр и площадь данного прямоугольника.

5. Даны координаты трех вершин треугольника: (x1, y1), (x2, y2), (x3, y3). Найти его периметр и площадь

**Блок-схемы**

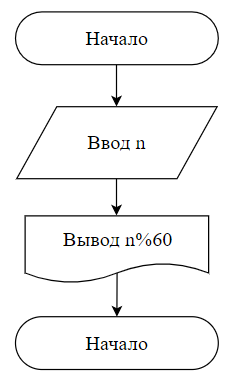


Рисунок 1 — Блок-схема к заданию 1

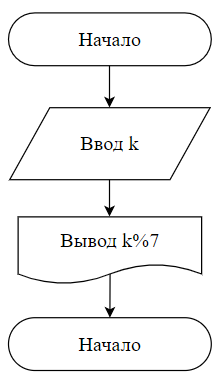
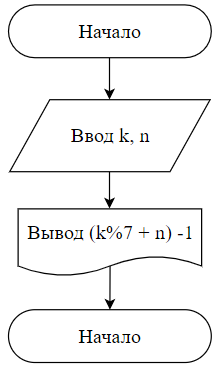


Рисунок 2 — Блок-схема к заданию 2   
  
   
  
Рисунок 3 — Блок-схема к заданию 3

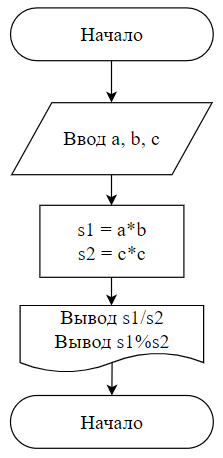
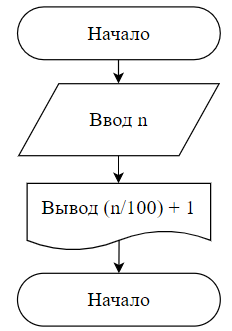


Рисунок 4 — Блок-схема к заданию 4   
  
   
  
Рисунок 5 — Блок-схема к заданию 5

# **Код программы**

Листинг 1 — Задание 1

|  |
| --- |
| 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 n; // создание переменных 12. n = int.Parse(Console.ReadLine()); 13. Console.WriteLine(n % 60); 14. Console.ReadLine(); // пауза для просмотра результата 15. } 16. } 17. } |

Листинг 2 — Задание 2

|  |
| --- |
| 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 k; // создание переменных 12. k = int.Parse(Console.ReadLine()); 14. Console.WriteLine(k % 7); 15. Console.ReadLine(); // пауза для просмотра результата 16. } 17. } 18. } |

|  |
| --- |
| 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 k, n; // создание переменных 12. k = int.Parse(Console.ReadLine()); 13. n = int.Parse(Console.ReadLine()); 14. Console.WriteLine((k % 7 + n) - 1); 15. Console.ReadLine(); // пауза для просмотра результата 16. } 17. } 18. } |

Листинг 3 — Задание 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, s1, s2; // создание переменных 12. a = int.Parse(Console.ReadLine()); 13. b = int.Parse(Console.ReadLine()); 14. c = int.Parse(Console.ReadLine()); 15. s1 = a \* b; s2 = c \* c; 16. Console.WriteLine("Number of squares included: " + s1 / s2); 17. Console.WriteLine("Unused area: " + s1 % s2); 18. Console.ReadLine(); // пауза для просмотра результата 19. } 20. } 21. } |

Листинг 4 — Задание 4

Листинг 5 — Задание 5

|  |
| --- |
| 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 n; // создание переменных 12. n = int.Parse(Console.ReadLine()); 13. Console.WriteLine("Decade of this year: " + ((n / 100) + 1)); 14. Console.ReadLine(); // пауза для просмотра результата 15. } 16. } 17. } |

# **Результат программы**

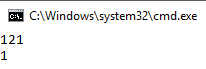


Рисунок 8 — Результат выполнения программы 1

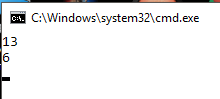


Рисунок 9 — Результат выполнения программы 2

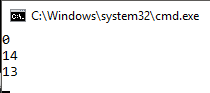


Рисунок 10 — Результат выполнения программы 3

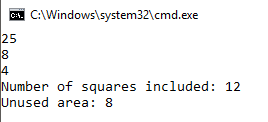


Рисунок 11 — Результат выполнения программы 4

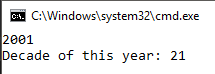


Рисунок 12 — Результат выполнения программы 5