**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 № 5**

**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](#_Toc21985922)

[**Задания** 7](#_Toc21985943)

[**Блок-схемы** 8](#_Toc21985944)

[**Код программы** 11](#_Toc21985945)

[**Результат программы** 14](#_Toc21985946)

# 

**Теория  
  
Арифметические операторы.**

## Следующие операторы выполняют арифметические операции с числовыми типами:

унарные — ++ (приращение), -- (уменьшение), + (плюс) и - (минус);

бинарные — \* (умножение), / (деление), % (остаток от деления), + (сложение) и - (вычитание).

# Эти операторы поддерживают все целочисленные типы и типы с плавающей запятой.

## Оператор инкремента ++

Оператор инкремента ++ увеличивает операнд на 1. Операндом должна быть переменная, свойство или индексатор.

Оператор инкремента поддерживается в двух формах: постфиксный оператор инкремента (x++) и префиксный оператор инкремента (++x).

# *Постфиксный оператор приращения*

Результатом x++ является значение x *перед* выполнением операции

# *Префиксный оператор инкремента*

Результатом ++x является значение x *после* выполнения операции

## Оператор декремента --

Унарный оператор декремента -- уменьшает операнд на 1. Операндом должна быть переменная, свойство или индексатор.

Оператор декремента поддерживается в двух формах: постфиксный оператор декремента (x--) и префиксный оператор декремента (--x).

# *Постфиксный оператор уменьшения*

Результатом x-- является значение x *перед* выполнением операции

# *Префиксный оператор декремента*

Результатом --x является значение x *после* выполнения операции

## Операторы унарного плюса и минуса

Унарный оператор + возвращает значение полученного операнда. Унарный оператор - изменяет знак операнда на противоположный.  
Унарный оператор - не поддерживает тип [ulong](https://docs.microsoft.com/ru-ru/dotnet/csharp/language-reference/builtin-types/integral-numeric-types).

## Оператор умножения \*

Оператор умножения \* вычисляет произведение операндов.  
Унарный оператор \* представляет собой оператор косвенного обращения к указателю.

## Оператор деления /

Оператор деления / делит левый операнд на правый.

# *Деление целых чисел*

Для операндов цельночисленных типов результат оператора / является целочисленным типом, который равен частному двух операндов, округленному в сторону нуля. Чтобы получить частное двух операндов в виде числа с плавающей запятой, используйте тип float, double или decimal:

# *Деление чисел с плавающей запятой*

Для типов float, double и decimal результатом оператора / является частное двух операндов. Если один из операндов — это decimal, второй операнд не может быть ни float, ни double, так как ни float, ни double не преобразуется неявно в тип decimal. Необходимо явным образом преобразовать операнд float или double в тип decimal. Дополнительные сведения о неявных числовых преобразованиях см. в таблице неявных числовых преобразований.

## Оператор остатка %

Оператор остатка % вычисляет остаток от деления левого операнда на правый.

# *Целочисленный остаток*

Для целочисленных операндов результатом a % b является значение, произведенное a - (a / b) \* b. Знак ненулевого остатка такой же, как и у левого операнда. Используйте метод Math.DivRem для вычисления результатов как целочисленного деления, так и определения остатка.

# *Остаток с плавающей запятой*

Для операндов типа float и double результатом x % y для конечных x и y будет значение z, так что:

знак z, если отлично от нуля, совпадает со знаком x;

абсолютное значение z является значением, произведенным |x| - n \* |y|, где n — это наибольшее возможное целое число, которое меньше или равно |x| / |y|, а |x| и |y| являются абсолютными значениями x и y, соответственно.

# *Примечание*

Этот метод вычисления остатка аналогичен тому, который использовался для целочисленных операндов, но отличается от IEEE 754. Если вам нужна операция остатка, которая соответствует IEEE 754, используйте метод **Math.IEEERemainder**.

Сведения о поведение оператора % в случае неконечных операндов см. в разделе Оператор остатка спецификации языка C#.

Для операндов decimal оператор остатка % эквивалентен оператору остатка типа System.Decimal.

## Оператор сложения +

Оператор сложения + вычисляет сумму своих операндов. Кроме того, оператор + можно использовать для объединения строк и делегатов.

## Оператор вычитания -

Оператор вычитания - вычитает правый операнд из левого. Кроме того, оператор - можно использовать для удаления делегатов.

## Приоритет и ассоциативность операторов

В следующем списке перечислены арифметические операторы в порядке убывания приоритета:

* Постфиксный инкремент x++ и декремент x--
* Префиксный инкремент ++x и декремент --x, унарные операторы + и -
* Мультипликативные операторы \*, /, и %
* Аддитивные операторы + и -

# **Задания**

1. Дан размер файла в байтах. Найти количество полных килобайтов, которые занимает данный файл

2. Даны целые положительные числа A и B (A > B). На отрезке длины A размещено максимально возможное количество отрезков длины B (без наложений). Найти количество отрезков B, размещенных на отрезке A.

3. Даны целые положительные числа A и B (A > B). На отрезке длины A размещено максимально возможное количество отрезков длины B (без наложений). Найти длину незанятой части отрезка A.

4. Дано двузначное число. Вывести число, полученное при перестановке цифр исходного числа.

5. Дано трехзначное число. В нем зачеркнули первую слева цифру и приписали ее справа. Вывести полученное число.

**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 (Найти количество полных килобайтов)

|  |
| --- |
| 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\_5\_\_1\_5\_ 7. { 8. class Program 9. { 10. static void Main(string[] args) 11. { 12. double B, Kb; 13. Console.Write("Введите размер файла данных в байтах: "); 14. B = double.Parse(Console.ReadLine()); 15. Kb = B / 1024; 16. Console.Write("Размер файла в килобайтах: " + Kb); 17. Console.ReadKey(); 18. } 19. } 20. } |

|  |
| --- |
| 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\_5\_\_1\_5\_ 7. { 8. class Program 9. { 10. static void Main(string[] args) 11. { 12. int A, B, P; 13. Console.WriteLine("Обязательное условие задачи: A>B"); 14. Console.Write("Введите длину отрезка A: "); 15. A = int.Parse(Console.ReadLine()); 16. Console.Write("Введите длину отрезка B: "); 17. B = int.Parse(Console.ReadLine()); 18. P = (A / B); 19. Console.Write("Количество отрезков B, размещенных на отрезке A: " + P); 20. Console.ReadKey(); 21. } 22. } 23. } |

Листинг 2 — Задание 2 (Найти количество отрезков B, размещенных на отрезке A)

Листинг 3 — Задание 3 (Найти длину незанятой части отрезка A)

|  |
| --- |
| 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\_5\_\_1\_5\_ 7. { 8. class Program 9. { 10. static void Main(string[] args) 11. { 12. int A, A1, B, P; 13. Console.WriteLine("Обязательное условие задачи: A>B"); 14. Console.Write("Введите длину отрезка A: "); 15. A = int.Parse(Console.ReadLine()); 16. Console.Write("Введите длину отрезка B: "); 17. B = int.Parse(Console.ReadLine()); 18. P = (A / B); 19. A1 = (A - P\*B); 20. Console.Write("Длина незанятой части отрезка A: " + A1); 21. Console.ReadKey(); 22. } 23. } 24. } |

|  |
| --- |
| 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\_5\_\_1\_5\_ 7. { 8. class Program 9. { 10. static void Main(string[] args) 11. { 12. int D, a, b; 13. Console.Write("Введите двузначное число: "); 14. D = int.Parse(Console.ReadLine()); 15. a = (D%10); 16. b = (D/10); 17. Console.Write("Новое число: " + a + b); 18. Console.ReadKey(); 19. } 20. } 21. } |

Листинг 4 — Задание 4 (Найти новое двузначное число)

Листинг 5 — Задание 5 (Найти новое трехзначное число)

|  |
| --- |
| 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\_5\_\_1\_5\_ 7. { 8. class Program 9. { 10. static void Main(string[] args) 11. { 12. int T, a, b; 13. Console.Write("Введите трехзначное число: "); 14. T = int.Parse(Console.ReadLine()); 15. a = (T % 100); 16. b = (T / 100); 17. Console.Write("Новое число: " + a + b); 18. Console.ReadKey(); 19. } 20. } 21. } |

# **Результат программы**

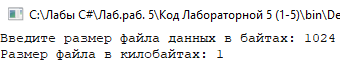


Рисунок 6 — Результат выполнения программы 1

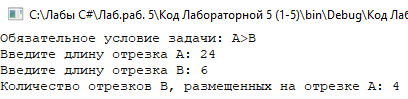


Рисунок 7 — Результат выполнения программы 2

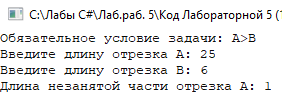


Рисунок 8 — Результат выполнения программы 3

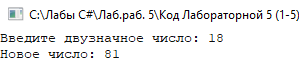


Рисунок 9 — Результат выполнения программы 4

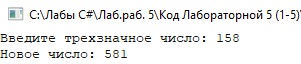


Рисунок 10 — Результат выполнения программы 5