



Task #06

# C# Basic Syntax. C# Class Static Methods and Fields. Constant Values. Linear Algorithms. Single Responsibility Principle, SRP



**LEARN. GROW. SUCCEED.**

© 2020. STEP Computer Academy - a leader in the field of professional computer education  
by Viktor Ivanchenko / [ivanvikvik@gmail.com](mailto:ivanvikvik@gmail.com) / Minsk

## Task #06

# Базовый синтаксис языка C#. Статические методы и поля класса в языке C#. Константные значения. Ли- нейные алгоритмы. Принцип единствен- ной ответственности

### Цель работы

Закрепить знания о примитивных (значимых) типах данных в C# и операциях, которые можно совершать над переменными и литералами данных типов; закрепить объявления и использования статических методов и полей класса, а также приобрести навыки решения задач с использованием линейных алгоритмов и принципа единственной ответственности на примере разработки простейших консольных C#-приложений.

### Требования

- 1) Для каждого задания в начале рекомендуется разработать блок-схему алгоритма решения.
- 2) При разработке программ рекомендуется придерживаться принципа единственной ответственности (**Single Responsibility Principle, SRP**), т.е. любой код (метод, класс и т.д.) должен быть настолько самостоятельным, насколько это возможно, чтобы его можно было повторно использовать в дальнейшем в других приложениях.
- 3) Если логически не подразумевается или в задании иного не указано, то входными и выходными данными являются вещественные числа (числа с плавающей запятой).
- 4) При написании кода считать, что пользователь вводит всегда корректные данные.
- 5) Целевые данные программы должны задаваться пользователем с консоли.

- 6) Для осуществления ввода-вывода данных с консоли рекомендуется использовать соответствующие статические методы класса **Console** из стандартного пространства имён **System**.
- 7) Программа должна быть снабжена дружелюбным и интуитивно понятным интерфейсом для взаимодействия с пользователем. Рекомендуется отображать интерфейс программы на английском языке.
- 8) При проверки работоспособности приложения необходимо проверить все тестовые случаи.
- 9) При разработке программ придерживайтесь соглашений по написанию кода на C# (C# *Code-Convention*).

## Общее задание

- 1) Разработайте программу нахождения периметра и площади прямоугольника с заданными сторонами  $a$  и  $b$ .
- 2) Разработайте программу нахождения среднего арифметического и среднего геометрического трёх неотрицательных чисел.
- 3) Разработайте программу нахождения максимального и минимального значения из двух чисел  $a$  и  $b$  (или из трёх чисел  $a$ ,  $b$  и  $c$ ).
- 4) На плоскости даны два круга с общим центром и радиусами  $R_1$  и  $R_2$  ( $R_1 > R_2$ ). Разработайте программу нахождения площади кольца, внешний радиус которого равен  $R_1$ , а внутренний радиус равен  $R_2$ .
- 5) Разработать программу решения линейного уравнение, заданного в виде  $Ax + B = 0$  (коэффициент  $A$  не равен 0).

## Дополнительное задание

- 1) Разработайте программу нахождения периметра и площади квадрата с заданной стороной  $a$ .
- 2) Разработайте программу нахождения площади и длины окружности по заданному её диаметру  $D$  (или её радиусу  $R$ ).
- 3) Разработайте программу нахождения площади объёма куба и площади его поверхности по заданному ребру  $a$ .
- 4) Разработайте программу нахождения гипотенузы прямоугольного треугольника, заданного сторонами  $a$  и  $b$ , а также периметр площадь данного треугольника.

- 5) Разработайте программу нахождения расстояния между двумя точками с заданными координатами  $(x_1, y_1)$  и  $(x_2, y_2)$  на плоскости.
- 6) Разработайте программу нахождения площади и периметра прямоугольника, который задан на координатной плоскости двумя противоположными вершинами  $(x_1, y_1)$  и  $(x_2, y_2)$ . Стороны прямоугольника параллельны осям координат.
- 7) Разработайте программу нахождения площади и периметра треугольника, который задан на координатной плоскости тремя вершинами:  $(x_1, y_1)$ ,  $(x_2, y_2)$ ,  $(x_3, y_3)$ .
- 8) Разработать программу вычисления пройденного пути лодкой, которая двигалась вначале по течению реки  $T_1$  ч., а потом против течения –  $T_2$  ч. Скорость лодки в стоячей воде  $V$  км./ч., скорость течения реки  $U$  км./ч. ( $U < V$ ).
- 9) Разработать программу вычисления расстояния между двумя автомобилями через  $T$  часов, если они движутся навстречу друг другу с соответствующими скоростями  $V_1$  км/ч и  $V_2$  км/ч, а первоначальное расстояние между ними  $S$  км.
- 10) Разработать программу вычисления расстояния между двумя автомобилями через  $T$  часов, если они удаляются друг от друга с соответствующими скоростями  $V_1$  км/ч и  $V_2$  км/ч, а первоначальное расстояние между ними  $S$  км.



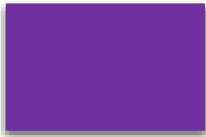

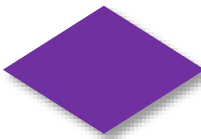

*Best of LUCK with it, and remember to HAVE FUN while you're learning :)*  
Victor Ivanchenko

## Графическое представление алгоритмов



Для общего представления решения задачи без привязки к конкретному языку программирования на практике используют блок-схемы. Они позволяют в графическом виде представить алгоритм решения задачи, который понятен не только разработчику, но даже домохозяйке.

Для графического представления алгоритмов решения задачи использую специальные унифицированные блоки, каждый из которых несёт в себе определённую смысловую нагрузку. Кратко каждый из наиболее востребованных блоков описывается в нижеприведённой таблице.

Таблица 1 – Наиболее часто используемые блоки

#	Shape (блок)	Description (описание)
1.		Блок начала/окончания выполнения программы
2.		Блок данных – используется для ввода, объявления и инициализации переменных программы
3.		Блок действия – используется для вычисления любых выражений программы
4.		Блок вызова процедур или функций – используется для обозначения вызова пользовательской функции или процедуры, код или реализация которой находится в другом файле
5.		Блок условия – задаёт соответствующие условия дальнейшего выбора хода выполнения кода программы
6.		Блок вывода данных – используется для обозначения выводимых данных или результата работы программы

Продолжение таблицы 1

7.		Блок соединитель на странице – используется в случае, если блок-схема алгоритма не может идти всё время сверху вниз и требуется её перенести на другую часть свободного места на том же листе
8.		Блок соединитель между страницами– используется в случае, если блок-схема алгоритма не помещается на одной странице и одну из её частей нужно перенести на другую страницу