Програмування

Рекомендована література

1. Албахари Д. C# 6.0. Справочник. Полное описание языка, 6-е изд. : Пер. с англ. / Джозеф Албахари, Бен Албахари. — М.: «И.Д. Вильямс», 2016. — 1040 с.
2. Вирт Н. Алгоритмы структуры данных, 2-е издание.: Пер. с англ. – СПб.: Невский Диалект, 2001. – 352 с.
3. Гриффитс И. Программирование на C# 5.0 : Пер. с англ. / Иэн Гриффитс. – М.: Эксмо, 2014. – 1136 с.
4. Нейгел К. C# 5.0 и платформа .NET4.5 для профессионалов / Нейгел К., Ивьен Б., Глинн Д., Уотсон К., Скиннер М. — М.: ООО «И.Д. Вильямс», 2014. — 1440 с.
5. Рихтер Дж. CLR via C#. Программирование на платформе Microsoft .NET Framework 4.5 на языке C#. / Дж. Рихтер. — Спб.: Питер, 2013. — 896 с.
6. Стэкер М. Разработка клиентских Windows-приложений на платформе Microsoft .NET Framework: Учебный курс Microsoft / Стэкер М., Стэйн С., Нортроп Т. — М.: Издательство «Русская редакция»; Спб.: Питер, 2008. — 624 с.
7. Троелсен Э. Язык программирования C# 5.0 и платформа .NET 4.5, 6-е изд. : Пер. с англ. / Эндрю Троелсен. – М.: ООО «И.Д. Вильямс», 2013. – 1312 с.

Основи програмування на C#

Приклад програми

```
using System;

namespace ConsoleApp1
{
	class Program
	{
		static void Main(string[] args)
		{
			string Text;
			Text = "I like C#!";
			Console.Write(Text);
			Console.ReadLine();
		}
	}
}
```

Основи програмування на C#

Оператори. Оператори в C# виконуються послідовно і завершуються крапкою з комою.

Оператори також можуть формувати блок коду.

Метод виконує дію у вигляді послідовності операторів, яка називається блоком операторів і являє собою пару фігурних дужок «{}», що містять нуль або більше операторів. Інші функції: конструктори, властивості, події, індексатори і фіналізатори.

Написання функцій вищого рівня, що викликають функції нижчих рівнів, спрощує написання програми.

Основи програмування на C#

Ідентифікатори – це імена, які програмісти обирають для своїх класів, методів, змінних та ін.

Ідентифікатор має бути суцільним словом та має починатися з літери або знаку підкреслення.

Ключові слова являють собою імена, що мають для компілятора особливий зміст. Більшість ключових слів зарезервовані, а це означає, що їх не можна використовувати в якості ідентифікаторів.

Якщо виникає необхідність застосувати ідентифікатор з іменем, що конфліктує з ключовим словом, тоді до ідентифікатора необхідно додати префікс @. Цей символ не є частиною самого ідентифікатора. Тобто, ідентифікатор @Text – це те ж саме, що й ідентифікатор Text.

Літерали – це елементарні порції даних, що лексично входять до програми.

У операторі присвоєння

```
Text = "I like C#!";
```

текст `I like C#!` є літералом.

Основи програмування на C#

Знаки пунктуації допомагають розмічати структуру програми. Наприклад:

{ } ;

Фігурні дужки {…} групують множину операторів у блок операторів.

Крапка з комою завершує оператор, але блоки операторів не вимагають в кінці крапки з комою.

Коментарі бувають в один рядок (відділяються подвійною нахиленою рискою //) і можуть розташовуватися на декількох рядках (починається символами /* і закінчується – */).

```
string text = "";
//Text = "";
/*Text = "";*/
Console.Write(Text);
```

Типи даних

Типи даних визначають шаблони для значень змінних або констант цього типу.

Наприклад:

```
int x=15; //оголошення змінної типу int
string T=”Це приклад змінної типу string”;
```

Складовими типу можуть бути дані і функції.

Дані певних типів створюються шляхом створення екземплярів цих типів. Створити екземпляр примітивного типу можна шляхом використання літералів, наприклад:

`”Це приклад літерала”` або `15`.

Класифікація типів

Класифікація типів у C#:

Типи значень

- Числовий
  - Цілі зі знаком (sbyte, short, int, long);
  - Цілі без знаку (byte, ushort, uint, ulong);
  - Дійсні (float, double, decimal);
- Логічний (bool);
- Символьний (char);

Типи посилань:

- рядок (string);
- масив (int[]);
- об’єкт (object).

Числові типи даних

Тип даних	Займає в пам’яті	Значення	Що зберігає
sbyte	1 байт	-128 .. 127	Знакове число
byte	1 байт	0 .. 255	Беззнакове число
short	2 байта	-32768 .. 32767	Знакове число
ushort	2 байта	0 .. 65535	Беззнакове число
int	4 байта	-2 147 483 648 .. 2 147 483 647	Знакове число
uint	4 байта	0 .. 4 294 967 295	Беззнакове число
long	8 байтів	-9 223 372 036 854 775 808 .. 9 223 372 036 854 775 807	Знакове число
ulong	8 байтів	0 .. 18 446 744 073 709 551 615	Беззнакове число
float	4 байта	1.5E-45 .. 3.4E+38	7 десяткових розрядів
double	8 байтів	|.| = 5E-324 .. 1.7E+308	15 десяткових розрядів
decimal	16 байтів	|.| = 1E-28 .. 7.9E+28	28 десяткових розрядів

Типи даних

Тип даних	Займає в пам’яті	Значення	Що зберігає
char	2 байта	`'\u0000' .. '\uffff'`	Символ Unicode
string		Обмежений системною пам’яттю набір символів Unicode	
bool		true, false	

Змінні

Змінні. Змінна позначає комірку пам’яті, яка із плином часу може містити різні значення. Всі значення в C# є екземплярами деякого типу. Смисл значення та допустимий набір можливих значень, яке здатна мати змінна, визначається її типом.

Змінна є зарезервоване місце в оперативній пам'яті для тимчасового зберігання даних. Кожна змінна має власне ім'я. Після того як змінній присвоєно значення, ви можете у програмі замість самого значення використовувати цю змінну.

Змінна позначає комірку пам’яті, яка із плином часу може містити різні значення. Всі значення в C# є екземплярами деякого типу. Смисл значення та допустимий набір можливих значень, яке здатна мати змінна, визначається її типом.

Оголошення змінних

Цілочисельні літерали можуть використовувати десяткову або шістнадцяткову форму запису; шістнадцяткова форма запису передбачає використання префіксу 0x. Наприклад:

```
int x=127;
int y=0x7F; //число 127 у шістнадцятковій формі
```

Дійсні літерали можуть використовувати десяткову або експоненціальну форму запису:

```
double a=1.5;
double million=1E06;
```

Числові суфікси

Для примусового визначення типу числових літералів можна використовувати числові суфікси, які можуть записуватися малими або великими літерами:

Суфікс	Тип C#	Приклад
F	float	float f=1.0F;
D	double	double d=1D;
M	decimal	decimal d=1.0M;
U	uint	uint i=1U;
L	long	long i=1L;
UL	ulong	ulong i=1UL;

Числові суфікси

Найкорисніші суфікси F і M. Їх завжди потрібно використовувати при вказуванні літералів float і decimal, оскільки без відповідних суфіксів числовий літерал із десятковою крапкою, наприклад 4.5, має тип double:

```
double d=4.5;   //компілюється без помилки
float f=4.5;    //буде помилка компіляції
float f=4.5F;   //компілюється без помилки
decimal D=4.5;  //буде помилка компіляції
decimal D=4.5M; //компілюється без помилки
```

Перетворення числових типів

Цілочисельні перетворення є **неявними**, якщо цільовий тип може представити кожне значення вихідного типу. В іншому випадку потрібно використовувати **явне** перетворення типу. Наприклад:

```
int x = 1024;
long y = x;
short z = (short)x;
```

Тип `float` може бути неявно перетвореним у тип `double`. Зворотне перетворення має бути явним.

Всі цілочисельні типи можуть бути неявно перетворені у числа із плаваючою крапкою. Зворотне перетворення має бути явним.

Всі цілочисельні типи можуть бути неявно перетворені у тип `decimal`. Всі інші числові перетворення у та із `decimal` мають бути явними.

Арифметичні операції

Арифметичні операції (+, –, *, /, %) визначені для всіх числових типів. Операції інкременту і декременту (++ і −−) збільшують або зменшують значення числових типів на 1.

Ця операція може знаходитися перед або після змінної в залежності від того, яке значення змінної нас цікавить – до або після виконання інкременту/декременту:

```
int x=0, y=0;

Console.WriteLine(x++);//виводить 0; x містить 1

Console.WriteLine(++y);//виводить 1; y містить 1
```