Променливи и работа с конзолата

1. Вход от конзолата с различни типове променливи

***Бележка: Може да копирате кода от всеки пример и да го поставите във Visual Studio, в направен от вас проект и да го изпълните с F5 или Ctrl + F5.***

* **string** променлива

Console.WriteLine("Please enter your name!");

string name = Console.ReadLine();

Console.WriteLine($"Your name is: {name}"); //Извеждане на екрана

* **int** променлива

Console.WriteLine("Please enter your age!");

int age = int.Parse(Console.ReadLine());

Console.WriteLine($"Your age is: {age}"); //Извеждане на екрана

* **long** променлива

Console.WriteLine("Please enter your id!");

long id = long.Parse(Console.ReadLine());

Console.WriteLine($"Your age is: {id}"); //Извеждане на екрана

* **float** променлива

Console.WriteLine("Please enter your salary!");

float floatSalary = float.Parse(Console.ReadLine());

Console.WriteLine($"Your salary is: {floatSalary}"); //Извеждане на екрана

* **double** променлива

Console.WriteLine("Please enter your salary!");

double salary = double.Parse(Console.ReadLine());

Console.WriteLine($"Your salary is: {salary}"); //Извеждане на екрана

**Parse** е метод (функция), който се вика, за да **конвертира** въведената стойност, към съответен тип: **int**, **long**, **double**, **float** и т.н.

1. Начини за извеждане на екрана на string-ове

public string string1 = "a";

public string string2 = "b";

public string string3 = "c";

public string string4 = "d";

public string string5 = "e";

* ‘+’ operator

Console.WriteLine(string1 + " " + string2 + " " + string3 + " " + string4 + " " + string5);

* String interpolation

Console.WriteLine($"{string1} {string2} {string3} {string4} {string5}");

* String format

Console.WriteLine(string.Format("{0} {1} {2} {3} {4}", string1, string2, string3, string4, string5));

***Бележка:***

***Има още начини за извеждането на стрингове на екрана на конзолата, но с тях ще се запознаем в по-късните етапи на обучението ни.***

1. Naming конвенции при именуването на променливи:

* CamelCase notation – **firstName**, **lastName**, **mySalary** и т.н.

***Важно:*** Характерното за този вид нотация е това, че първата дума на променливата се изписва с малка буква, а всяка следваща дума, която е част от името на променливата с голяма.

* PascalCase notation – **Engine**, **CarAttribute**, **DigitalInvertor** и т.н.

***Важно:*** Характерното за този вид нотация е това, че първата дума се изписва с голяма буква, както и всяка следваща дума, която е част от името.

* Hungarian notation – **strFirstName, iAge, bIsValid**

***Важно:*** Характерното за този вид нотация е това, че името на променливата, класа или която и да е езикова конструкция започва със съкращение, което показва типа на променливата.

Например в **strFirstName**, частта ‚str‘ показва, че променливата е от тип **string**.

***Важно:***

В **C#** за наименуване на променливи се използва **CamelCase** нотацията:

Примери за правилно наименувани променливи в C#:

* firstName
* age
* lastName
* isValid

За наименуване на класове, структури, методи, пропъртита се използва **PascalCase** нотацията:

Примери за правилно наименувани класове и методи в C#:

* CarEngine (клас)
* Person (клас)
* CalculateTrianglePerimeter (метод)
* CheckIfObjectExists (метод)

***Важно:***

В C# е препоръчително да не се използва **Hungarian** **notation**. За променливи се използва **CamelCase**, а за останалите **PascalCase**.

1. Имплицитно и Експлицитно кастване на променливи от различни типове

Може да се каже, че има два типа кастване:

* Имплицитно
* Експлицитно
* Имплицитно кастване имаме в следния случай:

int myIntNumber = 9;

long myLongNumber = myIntNumber;

Тук без проблем можем да присвоим променлива от тип **int** на променлива от тип **long**, защото **long** е много по-голям тип от **int**, в смисъл, че **long** типът има много по-голям диапазон от възможни стойности и това перфектно се разбира от **компилатора**.

* Експлицитно кастване имаме да кажем в следния случай:

float myFloatNumber = 9.2f;

int myIntNumber = myFloatNumber; //Не се компилира

Горният пример няма да се компилира и това може да се види във **Visual Studio**, защото средата ни казва, че ако искаме да присвоим стойност от тип **float**, на променлива от тип **int**, то може да има загуба на данни, а именно тези данни, които са зад **десетичната запетая**.

За да може да се компилира горния код трябва да направим малка промяна в него, която да каже на компилатора, че ние сме наясно със загубата на данни, която ще настъпи, ако направим това кастване.

float myFloatNumber = 9.2f;

int myIntNumber = (int)myFloatNumber; //Компилира се, но има загуба на данни