

# Типове данни и променливи

Низове. Обектен тип. Променливи

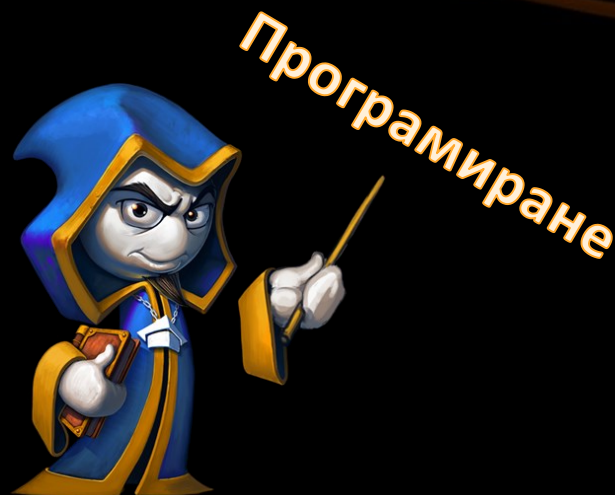


Учителски екип

Обучение за ИТ кариера

<https://it-kariera.mon.bg/e-learning/>

<https://github.com/BG-IT-Edu/School-Programming/tree/main/Courses/Applied-Programmer/Programming-Fundamentals>



# Съдържание

1. Низове

2. Обектен тип

3. Променливи

1. Именуване

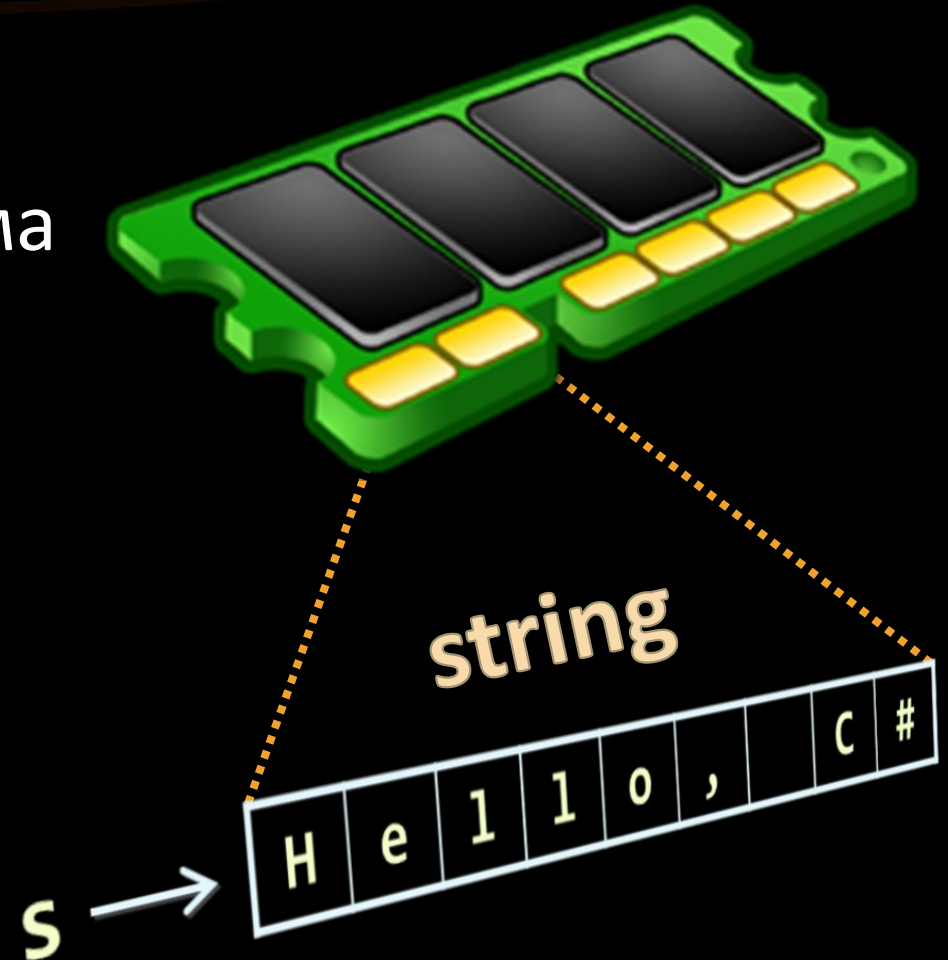
2. Живот и видимост

3. Промеждутък

# Низове

- Низовете в C#
  - Представят поредица от знаци
  - Задават се чрез **string** ключова дума
  - Имат стойност по подразбиране: **null** (празна стойност)
- Низовете се обграждат с кавички:

```
string s = "Hello, C#";
```
- Низовете могат да се слепят
  - Чрез оператор **+**



# Дословни (Verbatim) и съставни (Interpolated) низове

- Низовете са обградени от кавички "":

```
string file = "C:\\Windows\\win.ini";
```

Наклонената черта \  
се екранира с \\

- Низовете могат да са **дословни** (без екраниране):

```
string file = @"C:\Windows\win.ini";
```

Наклонената  
черта \ не се  
екранира

- Съставните** низове съдържат стойности на променливи по шаблон:

```
string firstName = "Svetlin";  
string lastName = "Nakov";  
string fullName = $"{firstName} {lastName}";
```



# Кажи „здрасти“ – Примери

- Комбиниране имената на човек, за да получите пълното име:

```
string firstName = "Ivan";  
string lastName = "Ivanov";  
Console.WriteLine(@"Hello, ""{0}""!", firstName);  
string fullName = $"{firstName} {lastName}";  
Console.WriteLine("Your full name is {0}.", fullName);
```

- Можем да слепим низовете с оператор +:

```
int age = 21;  
Console.WriteLine("Hello, I am " + age + " years old");
```

# Задача: Поздрав по име и възраст

- Напишете програма, която въвежда малкото име, фамилията и възрастта и извежда "Hello, <first name> <last name>. You are <age> years old."

```
string firstName = Console.ReadLine();  
string lastName = Console.ReadLine();  
string ageStr = Console.ReadLine();  
int age = int.Parse(ageStr); //Преобразуване string  
→ int  
  
Console.WriteLine($"Hello, {firstName}  
{lastName}.\r\nYou are {age} years old.");
```

# Обектен тип

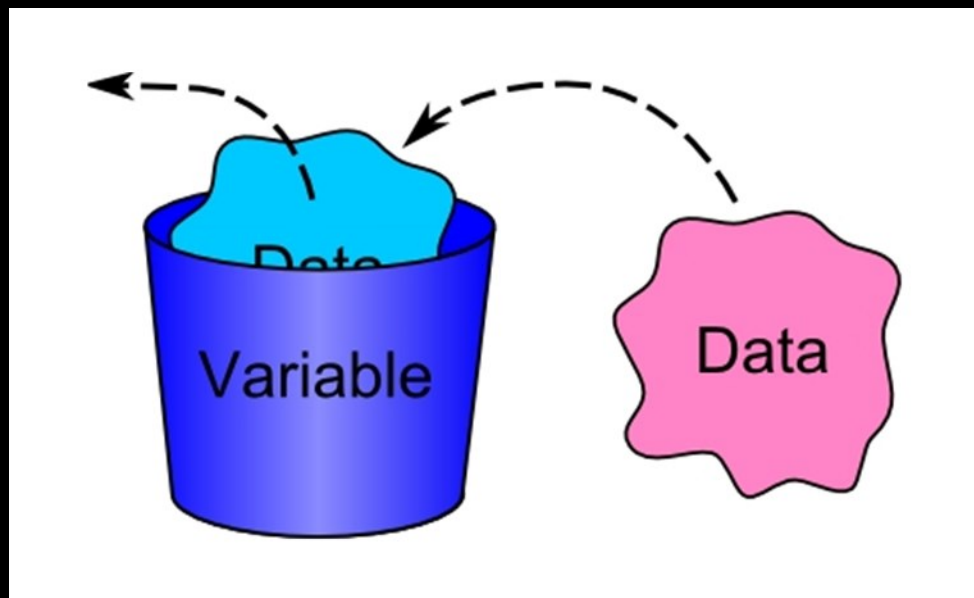
- Обектен тип в C#
  - Специален тип – родител на всички други типове в .NET
  - Задава се чрез **object** ключова дума
  - Може да приема стойности, от които и да е тип
  - Референтен тип – съдържа указател към област в паметта, на която се съхранява неговата стойност

# Задача: Низове и обекти

- Декларирайте две **string** променливи и им задайте стойности
- В променлива от тип **object** присвоете резултата от слепянето на двете променливи
- От там прехвърлете на друга променлива от тип **string**

```
string a = "Hello";  
string b = "World";  
object c = a + " " + b;  
string d = (string) c;  
Console.WriteLine(d);
```



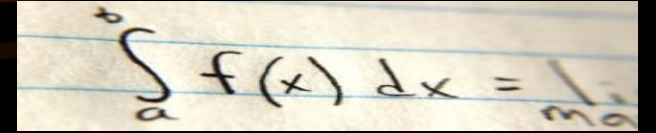


# Променливи

# Именуване на променливи

- Имената на променливите

- Винаги използвайте **конвенциите** за именуване на даден програмен език – за C# ползвайте **camelCase**
- Предпочитан формат: [съществително] или [прилагателно] + [съществително]
- Трябва да описва **предназначението** на **променливата** (Винаги се питайте „Какво съдържа тази променлива?“)


$$\int_a^b f(x) dx = \frac{1}{30}$$



`firstName, report, config, userList, fontSize, maxSpeed`



`foo, bar, p, p1, p2, populate, LastName, last_name, LAST_NAME`

# Живот и област на видимост на променливите

- Област на видимост (variable scope) == мястото където можем да достъпим променлива (глобално, локално)
- Живот (variable lifetime) == колко дълго остава в паметта

```
static void Main()
{
    var outer = "I'm inside the Main()";
    for (int i = 0; i < 10; i++)
    {
        var inner = "I'm inside the loop";
    }
    Console.WriteLine(outer);
    // Console.WriteLine(inner); // Грешка
}
```

Достъпна в **Main()**

Достъпни в цикъла

# Промеждутък на променлива

- Промеждутък (*variable span*) определя колко време съществува една променлива преди да я използваме
- Винаги създавайте променливата колкото се може **по-късно**

```
static void Main()
{
    var outer = "I'm inside the Main()";
    for (int i = 0; i < 10; i++)
    {
        var inner = "I'm inside the loop";
    }
    Console.WriteLine(outer);
    // Console.WriteLine(inner); // Грешка
}
```

Промеждутък  
на "**outer**"



# Поддържайте кратък промеждутък

- По-краткия промеждутък опростява кода
  - Подобрява неговата четимост и улеснява бъдещи промени

```
static void Main()
{
    for (int i = 0; i < 10; i++)
    {
        var inner = "I'm inside the loop";
    }
    var outer = "I'm inside the Main()";
    Console.WriteLine(outer);
    // Console.WriteLine(inner); // Error
}
```

Промеждутъкът  
на "outer" –  
намален

# Задача: Рефакториайте кода

- Имате работещ код за намиране на обема на пирамида:
  - Оправете именуването, промеждутъка и използването на променливите:

```
double du1, sh, V = 0;  
Console.Write("Length: ");  
du1 = double.Parse(Console.ReadLine());  
Console.Write("Width: ");  
sh = double.Parse(Console.ReadLine());  
Console.Write("Height: ");  
V = double.Parse(Console.ReadLine());  
V = (du1 * sh * V) / 3;  
Console.WriteLine("Pyramid Volume: {0:F2}", V);
```

# Задача: Рефакторируйте Специални числа

```
int kolkko = int.Parse(Console.ReadLine());
int obshto = 0; int takova = 0; bool toe = false;
for (int ch = 1; ch <= kolkko; ch++)
{
    takova = ch;
    while (ch > 0)
    {
        obshto += ch % 10;
        ch = ch / 10;
    }
    toe = (obshto == 5) || (obshto == 7) || (obshto == 11);
    Console.WriteLine($"{takova} -> {toe}");
    obshto = 0; ch = takova;
}
```

string

bool

char



(int) value

switch  
case

# Променливи

## Експерименти



# Какво научихме днес?

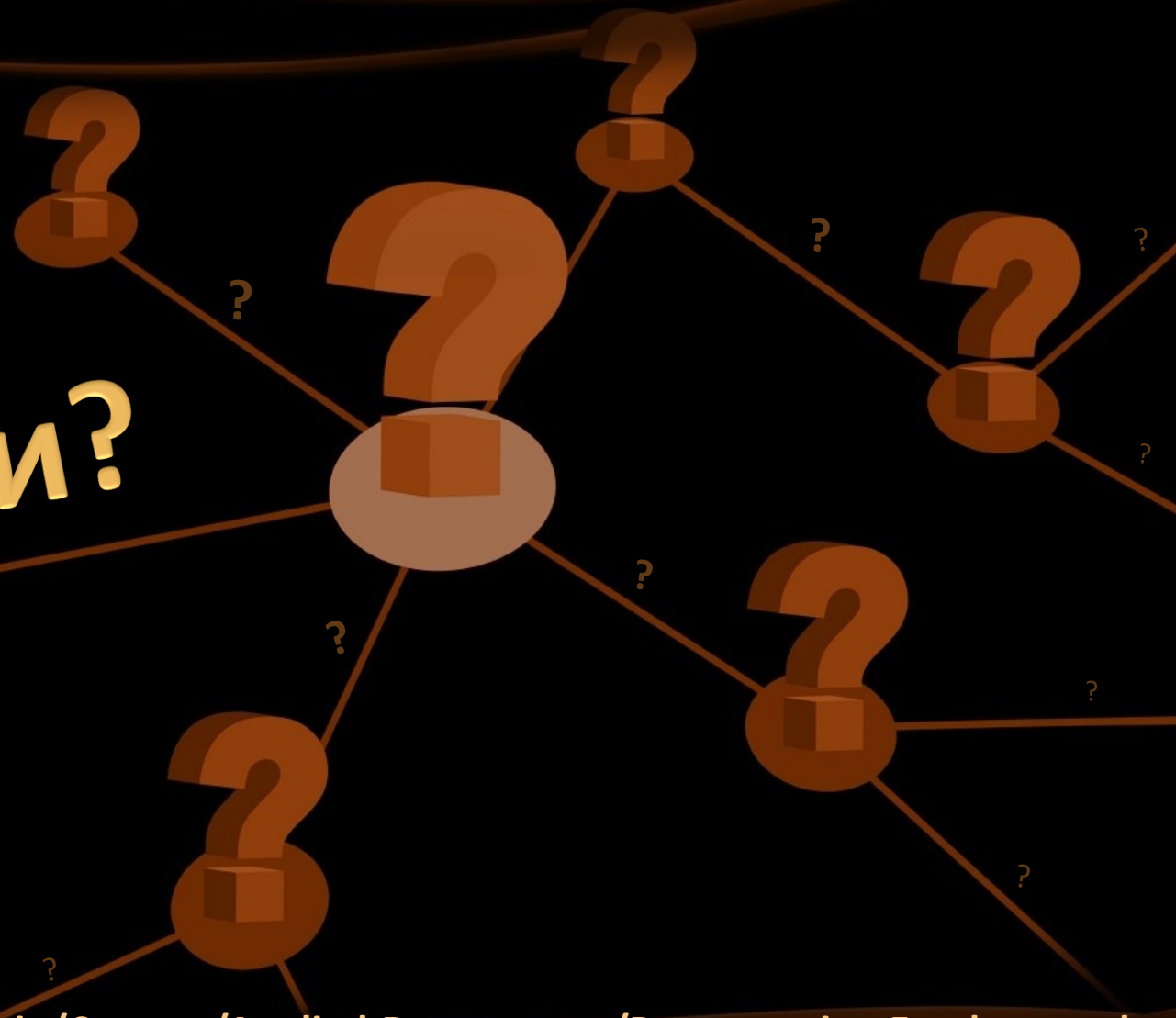
- Класически типове данни:
  - Низове: съдържат **текст**
  - Поредици от **Уникод** знаци
- Обектен тип
  - Може да приема стойности от всеки друг тип
  - Представява указател към област в паметта
- Променливи – съдържат **информация**
  - Трябва да се именува добре, да се намаля тяхната **област, промеждутък и живот**



# Типове данни и променливи



Въпроси?



# Министерство на образованието и науката (МОН)

- Настоящият курс (презентации, примери, задачи, упражнения и др.) е разработен за нуждите на Национална програма **"Обучение за ИТ кариера"** на МОН за подготовка по професия "Приложен програмист"



Министерство  
на образованието  
и науката



Национална  
програма  
„Обучение за  
ИТ кариера“

- Курсът е базиран на учебно съдържание и методика, предоставени от **фондация "Софтуерен университет"** и се разпространява под свободен лиценз **CC-BY-NC-SA**



SoftUni  
Foundation

