

C# Introduction

Basic Syntax , I/O, Conditions, Loops and Debugging



Software University
Technical Trainers



SoftUni



Software University

<https://softuni.bg>

Table of Contents

1. Introduction and Basic Syntax
2. Input / Output
3. Comparison Operators
4. Implementing Control-Flow Logic
5. Logical Operators
6. Loops
7. Debugging and Troubleshooting



sli.do

#fund-csharp



Introduction and Basic Syntax

C# Programming Language

- **C#** is modern, flexible, general-purpose programming language
- **Object-oriented** by nature, statically-typed, compiled
- Runs on .NET Framework / .NET Core

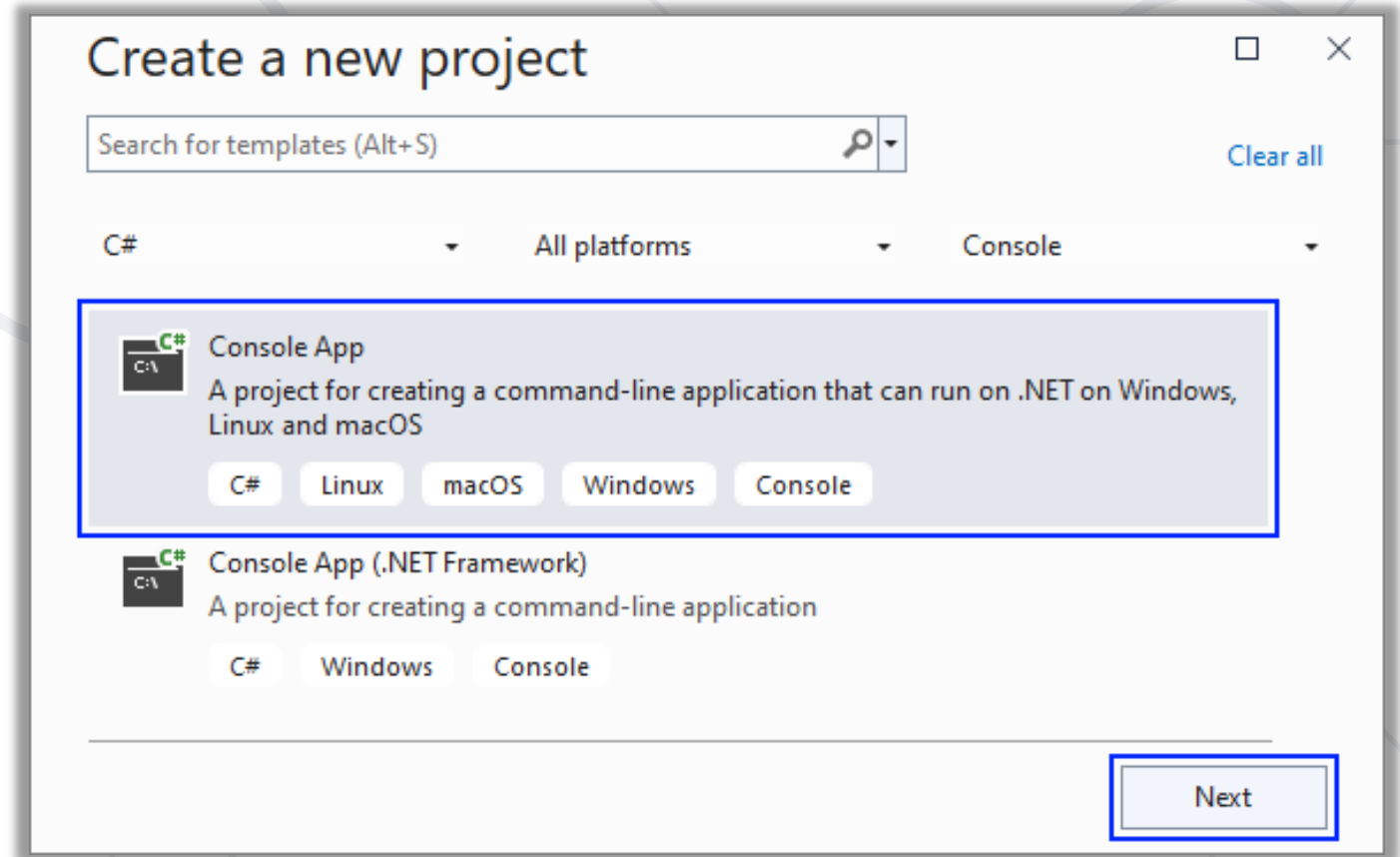


```
static void Main()  
{  
    // Source code  
}
```

Program
starting
point


Using Visual Studio (1)

- **Visual Studio** (VS) is powerful IDE for C#
- Create a **console application**



Using Visual Studio (2)

- Give the console application a **proper name**



Configure your new project

Console App C# Linux macOS Windows Console

Project name
HelloCSharp

Location
C:\Repos

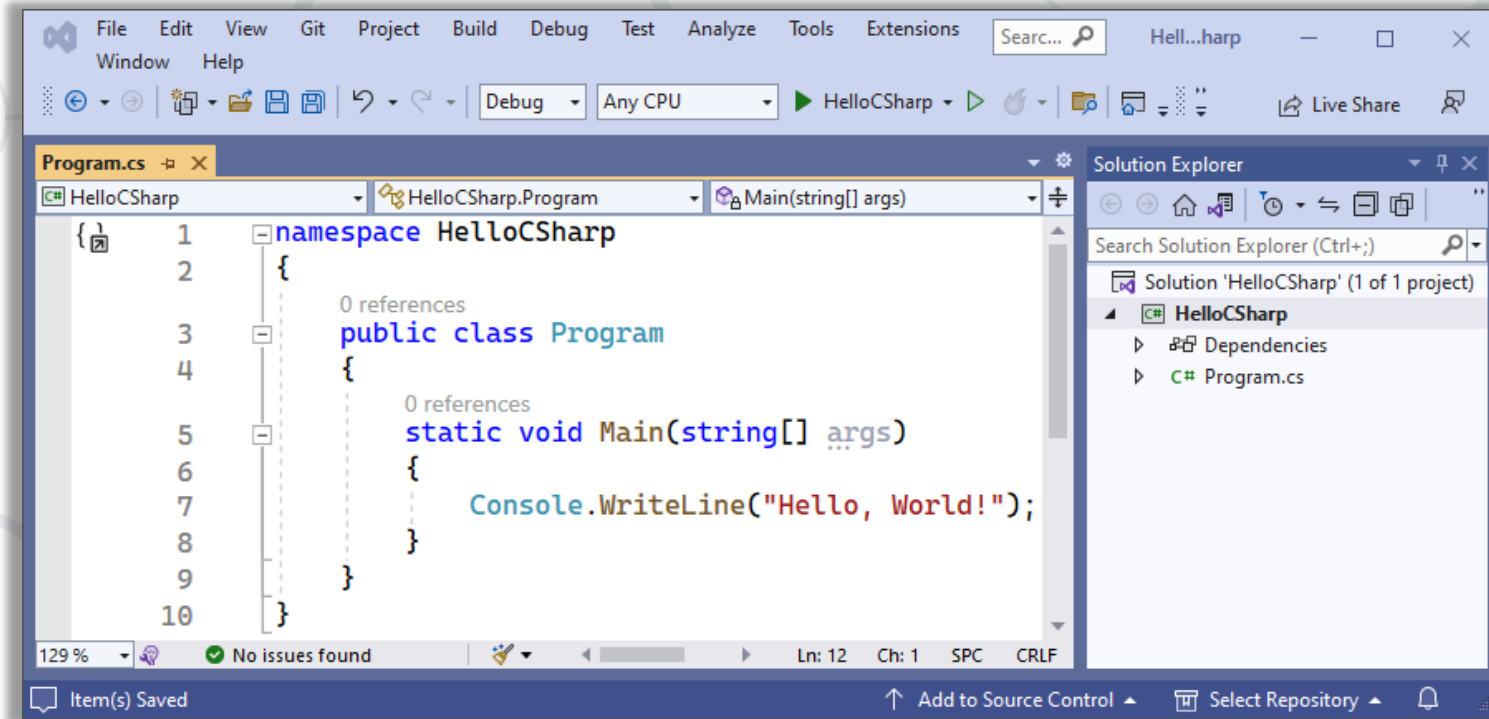
Solution name ⓘ
HelloCSharp

☐ Place solution and project in the same directory

Back Next

Running the Program

- Start the program from VS using [Ctrl + F5]



```
1 namespace HelloCSharp
2 {
3     0 references
4     public class Program
5     {
6         0 references
7         static void Main(string[] args)
8         {
9             Console.WriteLine("Hello, World!");
10        }
11    }
```



Declaring Variables

- Defining and Initializing variables

```
{data type / var} {variable name} = {value};
```

- Example

```
int number = 5;
```

Variable name

Data type

Variable value





Input / Output

Reading from and Writing to the Console

Reading from the Console

- We can **read/write** to the console, using the **Console** class
- Use the **System** namespace to access **System.Console** class



```
using System;
```

- Reading input from the console using **Console.ReadLine()**

Returns **string**

```
string name = Console.ReadLine();
```

Converting Input from the Console

- `Console.ReadLine()` returns a **string**
- Convert the string to number by **parsing**

```
string name = Console.ReadLine();  
int age = int.Parse(Console.ReadLine());  
double salary = double.Parse(Console.ReadLine());  
bool isHungry = bool.Parse(Console.ReadLine());
```



- We can **print** to the console using the **Console** class
- Use the **System** namespace to access **System.Console** class
- Writing output to the console
 - **Console.Write()**
 - **Console.WriteLine()**

```
Console.Write("Hi, ");  
Console.WriteLine("John!");  
// Hi, John!
```

- Using **placeholders** to print on the console
- Examples

```
string name = "George";  
int age = 5;  
Console.WriteLine("Name: {0}, Age: {1}", name, age);  
// Name: George, Age: 5
```

Placeholder **{0}**
corresponds to **name**

Placeholder **{1}**
corresponds to **age**

Formatting Numbers in Placeholders

- **D** – format number to certain digits with leading zeros
- **F** – format floating point number with certain digits after the decimal point
- Examples

```
double grade = 5.5334;  
int percentage = 55;  
Console.WriteLine("{0:F2}", grade);           // 5.53  
Console.WriteLine("{0:D3}", percentage);       // 055
```

- Using string interpolation to print on the console
- Examples

```
string name = "George";  
int age = 5;
```

Put **\$** in front of the
string to use **string
interpolation**

```
Console.WriteLine($"Name: {name}, Age: {age}");  
// Name: George, Age 5
```


Problem: Student Information

- You will be given 3 input lines:
 - Student Name, Age and Average Grade
- Print the input in the following format:
 - "Name: {name}, Age: {age}, Grade: {grade}"
 - Format the grade to 2 decimal places

John
15
5.40

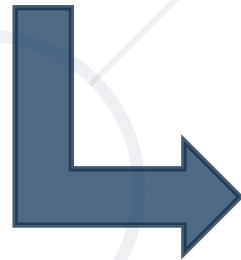


Name: John, Age: 15, Grade: 5.40

Check your solution here: <https://judge.softuni.org/Contests/Practice/Index/1188#0>

Solution: Student Information

```
string name = Console.ReadLine();  
int age = int.Parse(Console.ReadLine());  
double grade = double.Parse(Console.ReadLine());  
  
Console.WriteLine($"Name: {name}, Age: {age}, Grade: {grade:f2}");
```



Name: John, Age: 15, Grade: 5.40



Comparison Operators

Comparison Operators

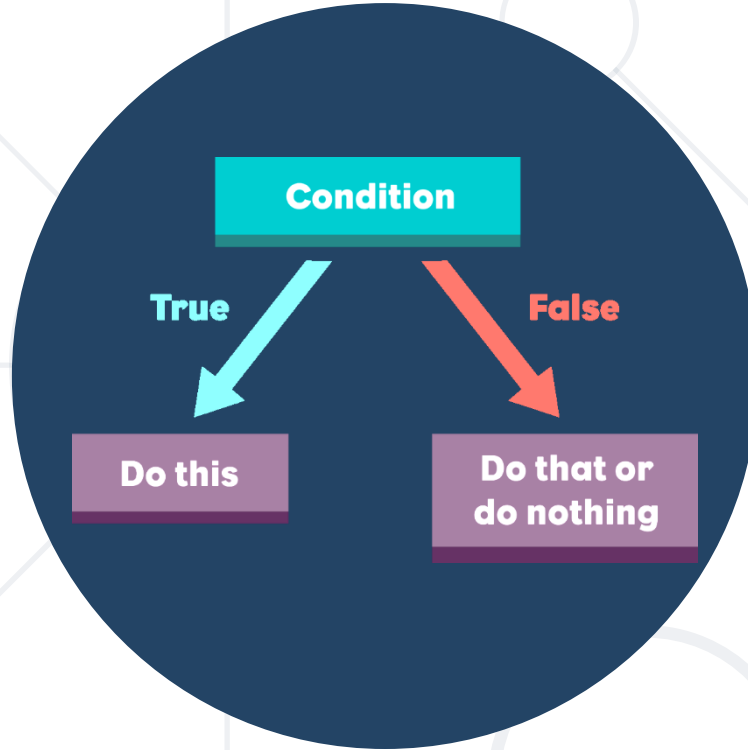
Operator	Notation in C#
Equals	<code>==</code>
Not Equals	<code>!=</code>
Greater Than	<code>></code>
Greater Than or Equals	<code>>=</code>
Less Than	<code><</code>
Less Than or Equals	<code><=</code>

Comparing Numbers

- Values can be compared:

```
int a = 5;
int b = 10;

Console.WriteLine(a < b);           // true
Console.WriteLine(a > 0);           // true
Console.WriteLine(a > 100);         // false
Console.WriteLine(a < a);           // false
Console.WriteLine(a <= 5);          // true
Console.WriteLine(b == 2 * a);      // true
```



Implementing Control-Flow Logic

The If-else Statement

- The most simple conditional statement
 - Test for a condition
- Example: Take as an input a grade and check if the student has passed the exam (grade ≥ 3.00)

```
double grade = double.Parse(Console.ReadLine());  
if (grade >= 3.00)  
{  
    Console.WriteLine("Passed!");  
}
```

In C# the opening bracket stays on a new line

Check your solution here: <https://judge.softuni.org/Contests/Practice/Index/1188#1>

The If-Else Statement

- Executes **one branch** if the condition is **true** and **another** if it is **false**
- Example: **Upgrade** the last example, so it prints "**Failed!**" if the mark is lower than 3.00:

The **else** keyword stays on a new line

```
if (grade >= 3.00)
{
    Console.WriteLine("Passed!");
}
else
{
    // TODO: Print the message
}
```

Check your solution here: <https://judge.softuni.org/Contests/Practice/Index/1188#2>

Problem: Back in 30 Minutes

- Write a program that reads hours and minutes from the console and calculates the time after 30 minutes
 - The hours and the minutes come on separate lines
- Examples

1
46 → 2:16

0
01 → 0:31

23
59 → 0:29

11
08 → 11:38

12
49 → 13:19

11
32 → 12:02

Check your solution here: <https://judge.softuni.org/Contests/Practice/Index/1188#3>

Solution: Back in 30 Minutes

```
int hours = int.Parse(Console.ReadLine());
int minutes = int.Parse(Console.ReadLine()) + 30;
if (minutes > 59) {
    hours += 1;
    minutes -= 60;
}
if (hours > 23) {
    hours = 0;
}
Console.WriteLine("{0}:{1:D2}", hours, minutes);
```

Check your solution here: <https://judge.softuni.org/Contests/Practice/Index/1188#3>



The Switch-Case Statement

Simplified If-else-if-else

The Switch-case Statement

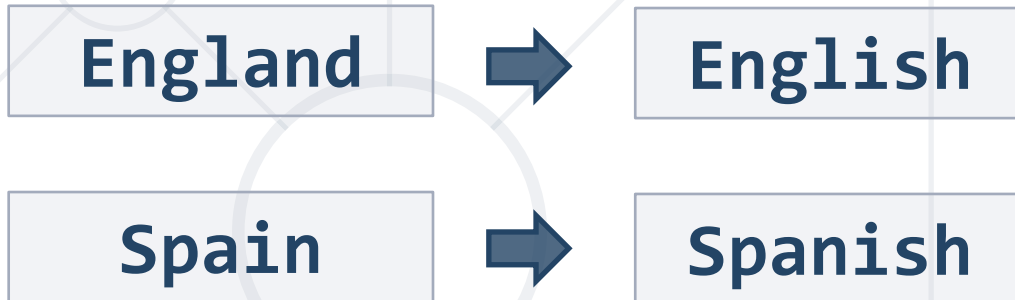
- Switch-case statement works as a sequence of **if-else**
- Example: Read an input number and print its corresponding month

```
int month = int.Parse(Console.ReadLine());  
switch (month)  
{  
    case 1: Console.WriteLine("January"); break;  
    case 2: Console.WriteLine("February"); break;  
    // TODO: Add the other cases  
    default: Console.WriteLine("Error!"); break;  
}
```

Check your solution here: <https://judge.softuni.org/Contests/Practice/Index/1188#4>

Problem: Foreign Languages

- By given country print its typical language:
 - English → England, USA
 - Spanish → Spain, Argentina, Mexico
 - other → unknown



Check your solution here: <https://judge.softuni.org/Contests/Practice/Index/1188#5>

Solution: Foreign Languages

```
switch (country)
{
    case "USA":
    case "England": Console.WriteLine("English"); break;
    case "Spain":
    case "Argentina":
    case "Mexico": Console.WriteLine("Spanish"); break;
    default: Console.WriteLine("unknown"); break;
}
```

Check your solution here: <https://judge.softuni.org/Contests/Practice/Index/1188#5>

A background network diagram consisting of a grid of light gray lines intersecting at various points. Some of these intersections are marked with small, light gray circles. A larger, solid dark blue circle is centered in the upper half of the image, containing the text '&&'.

&&

Logical Operators

Writing More Complex Conditions

- Logical operators give us the ability to write multiple conditions in one **if** statement
- They return a boolean value and compare boolean values

Operator	Notation in C#	Example
Logical NOT	!	!false → true
Logical AND	&&	true && false → false
Logical OR		true false → true

Problem: Theatre Promotions

- A theatre has the following ticket prices according to the age of the visitor and the type of day
 - If the age is < 0 or > 122 , print "Error!":

Day / Age	$0 \leq \text{age} \leq 18$	$18 < \text{age} \leq 64$	$64 < \text{age} \leq 122$
Weekday	12\$	18\$	12\$
Weekend	15\$	20\$	15\$
Holiday	5\$	12\$	10\$

Weekday
42



18\$

Holiday
-12



Error!

Check your solution here: <https://judge.softuni.org/Contests/Practice/Index/1188#6>

Solution: Theatre Promotions

```
var day = Console.ReadLine().ToLower();
var age = int.Parse(Console.ReadLine());
var price = 0;
if (day == "weekday")
{
    if ((age >= 0 && age <= 18) || (age > 64 && age <= 122))
    {
        price = 12;
    }
    // TODO: Add else statement for the other group
} // Continues on the next slide...
```

Solution: Theatre Promotions (2)

```
else if (day == "weekend")
{
    if ((age >= 0 && age <= 18) || (age > 64 && age <= 122))
    {
        price = 15;
    }
    else if (age > 18 && age <= 64)
    {
        price = 20;
    }
} // Continues on the next slide...
```

Solution: Theatre Promotions (3)

```
else if (day == "holiday")
{
    if (age >= 0 && age <= 18)
        price = 5;
    // TODO: Add the statements for the other cases
}
if (price != 0)
    Console.WriteLine(price + "$");
else
    Console.WriteLine("Error!");
```

Check your solution here: <https://judge.softuni.org/Contests/Practice/Index/1188#6>



Loops

Code Block Repetition

Loop: Definition

- A loop is a control statement that repeats the execution of a block of statements. The loop can
 - Execute a code block a fixed number of times
 - **for** loop
 - Execute a code block while a given condition returns true
 - **while**
 - **do...while**





For Loops

Managing the Count of the Iteration

- The **for** loop executes statements a fixed number of times:

Initial value

End value

Increment

The bracket
is again at
the new line

```
for (int i = 1; i <= 10; i++)  
{  
    Console.WriteLine("i = " + i);  
}
```

Loop **body**,
executed
each
iteration

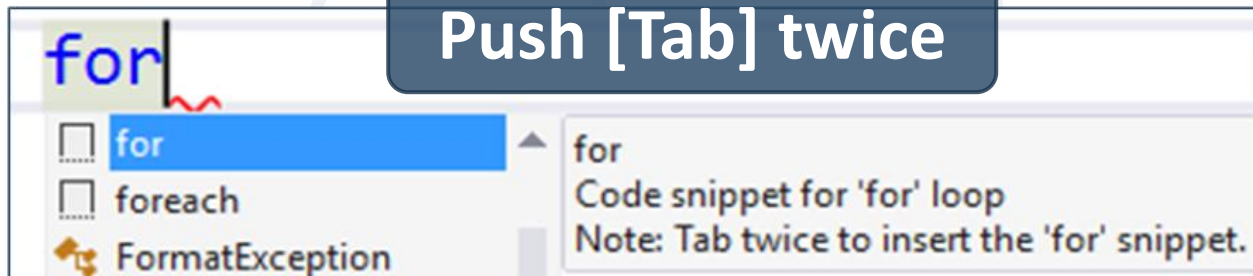
Example: Divisible by 3

- Print the numbers from 1 to 100 that are divisible by 3

```
for (var i = 3; i <= 100; i += 3)
{
    Console.WriteLine(i);
}
```



- You can use "for" code snippet in Visual Studio

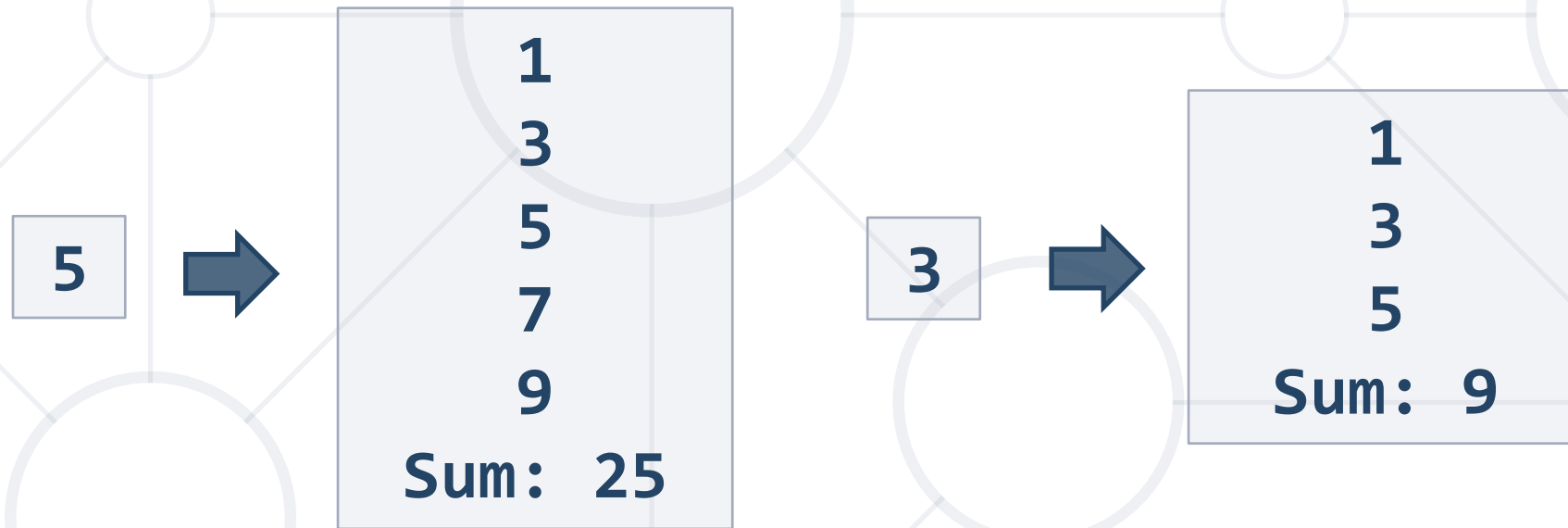


```
for (int i = 0; i < length; i++)
{
    ...
}
```

Check your solution here: <https://judge.softuni.org/Contests/Practice/Index/1188#7>

Problem: Sum of Odd Numbers

- Write a program to print the first **n** odd numbers and their sum



Check your solution here: <https://judge.softuni.org/Contests/Practice/Index/1188#8>

Solution: Sum of Odd Numbers

```
var n = int.Parse(Console.ReadLine());  
var sum = 0;  
  
for (int i = 1; i <= n; i++)  
{  
    Console.WriteLine("{0}", 2 * i - 1);  
    sum += 2 * i - 1;  
}  
  
Console.WriteLine("Sum:{0}", sum);
```

Check your solution here: <https://judge.softuni.org/Contests/Practice/Index/1188#8>



Iterations While a Condition is True

While Loops

While Loops

- Executes commands **while** the **condition** is **true**

Initial value

```
var n = 1;
```

```
while (n <= 10)
```

Condition

```
{
```

```
    Console.WriteLine(n);
```

Loop body

```
    n++;
```

Increment the counter

```
}
```

Problem: Multiplication Table

- Print a table holding $\text{number} \times 1$, $\text{number} \times 2$, ..., $\text{number} \times 10$

```
var number = int.Parse(Console.ReadLine());  
var times = 1;  
while (times <= 10)  
{  
    Console.WriteLine(  
        $"{number} X {times} = {number * times}");  
    times++;  
}
```

Check your solution here: <https://judge.softuni.org/Contests/Practice/Index/1188#9>

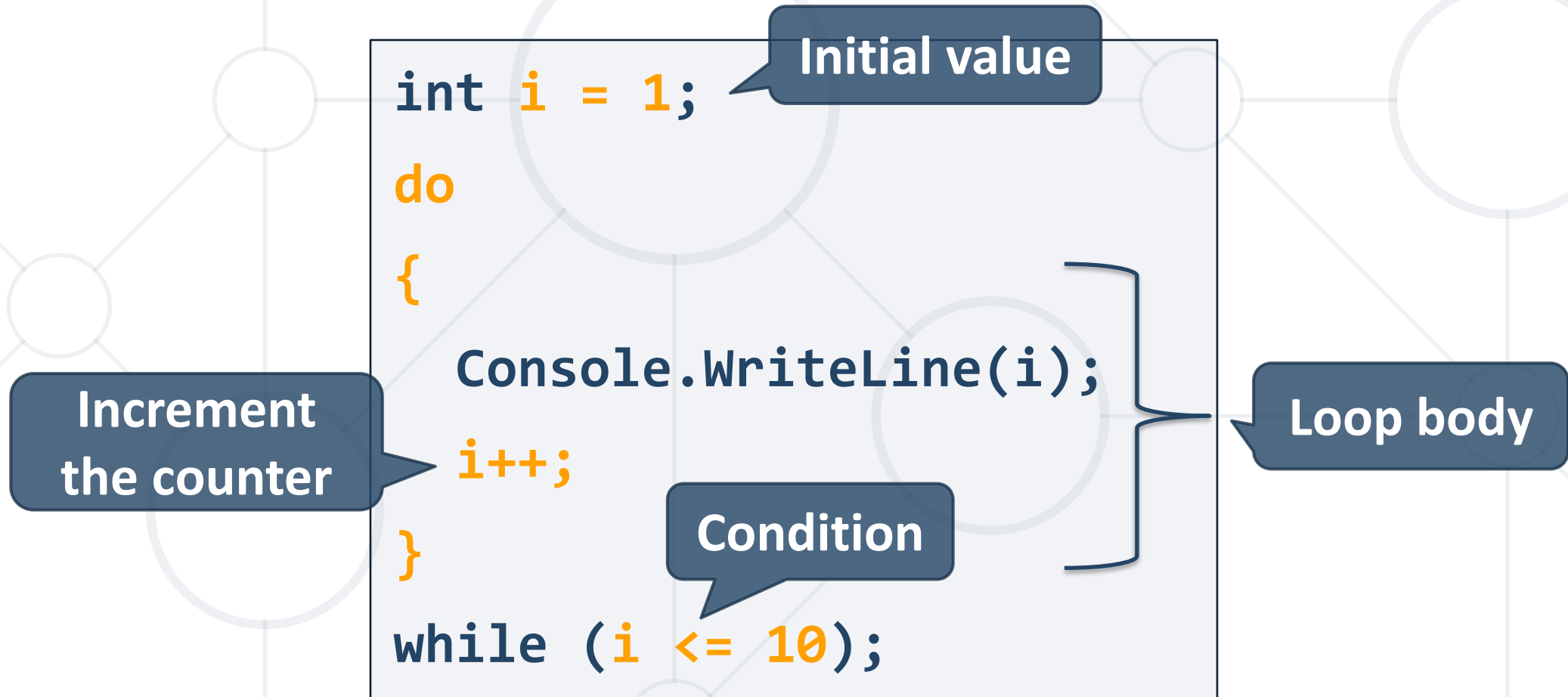


Do...While Loop

Executes Code Block One or More Times

Do...While Loop

- Similar to the **while** loop, but always executes at least once



Problem: Multiplication Table 2.0

- Upgrade your program and take the initial times from the console

```
int number = int.Parse(Console.ReadLine());
int times = int.Parse(Console.ReadLine());
do
{
    Console.WriteLine(
        $"{number} X {times} = {number * times}"
    );
    times++;
} while (times <= 10);
```

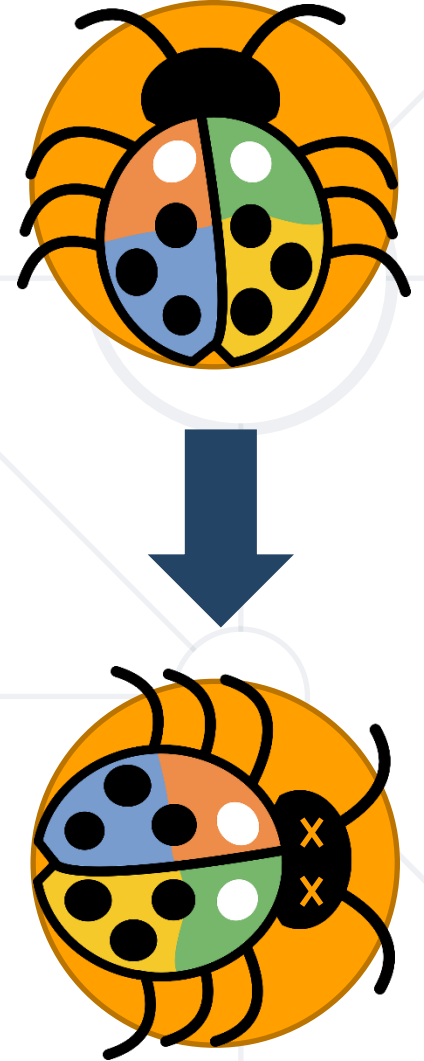
Check your solution here: <https://judge.softuni.org/Contests/Practice/Index/1188#10>



Debugging and Troubleshooting

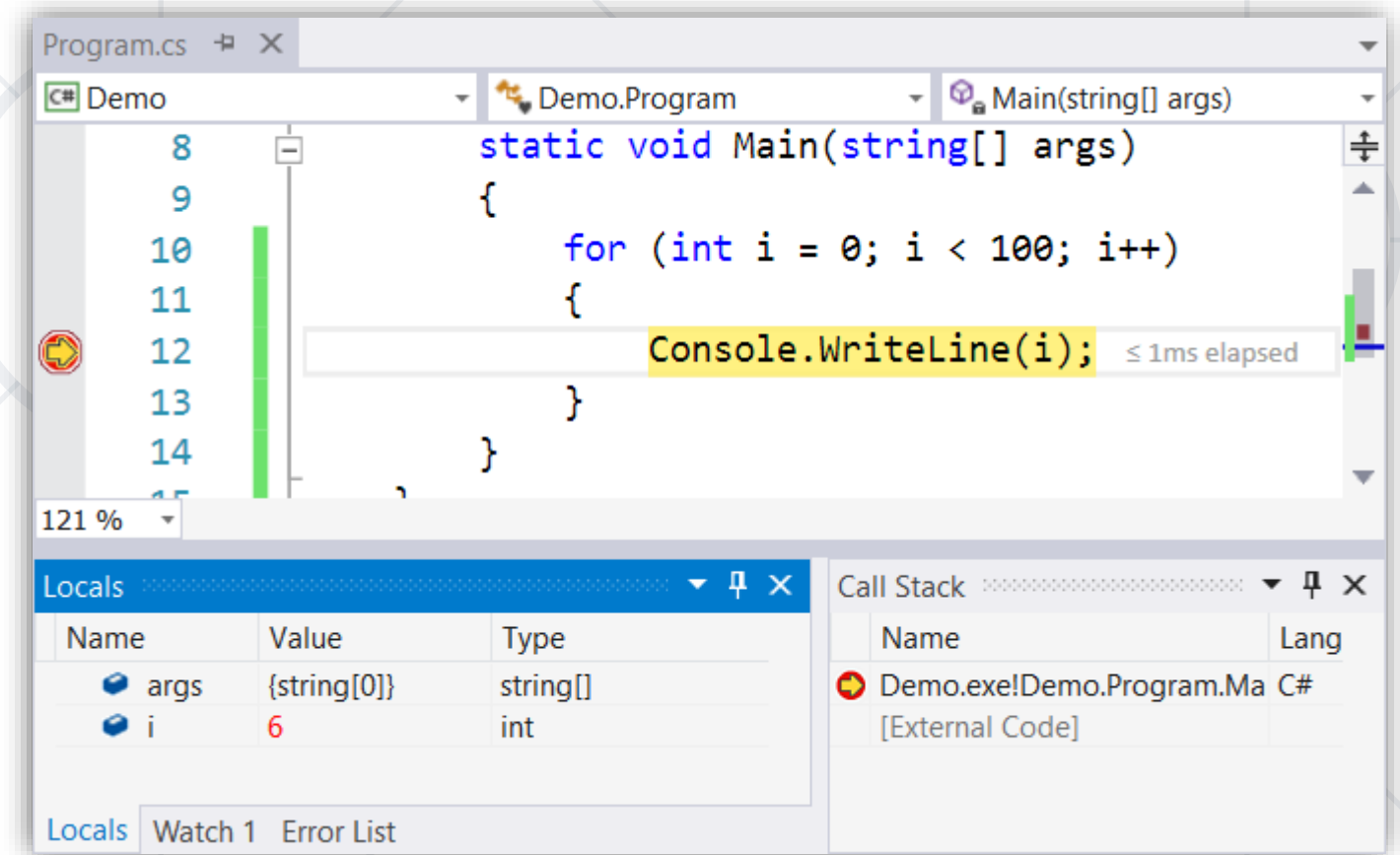
Using the Visual Studio Debugger

- The process of **debugging an application** includes
 - Spotting an error
 - Finding the lines of code that cause the error
 - Fixing the error in the code
 - Testing to check if the error is gone and no new errors are introduced
- Iterative and continuous process



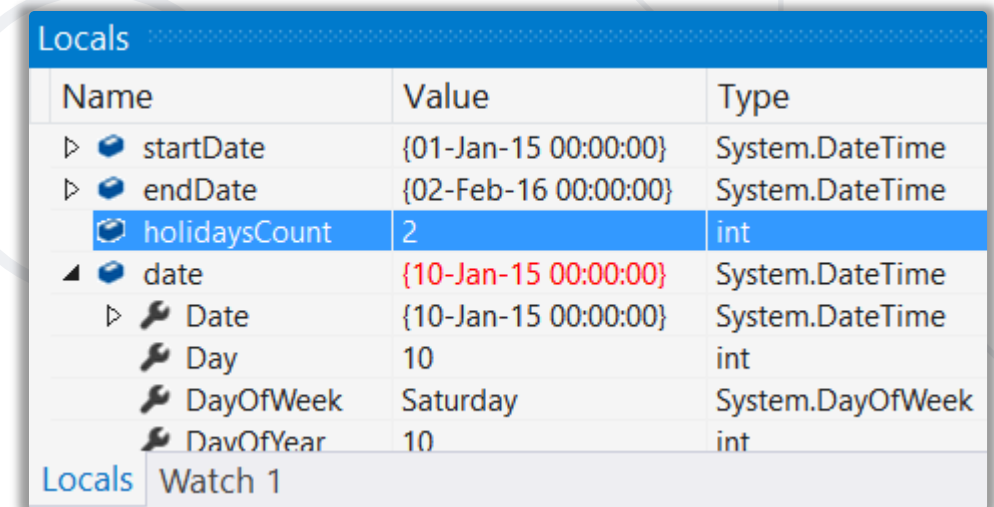
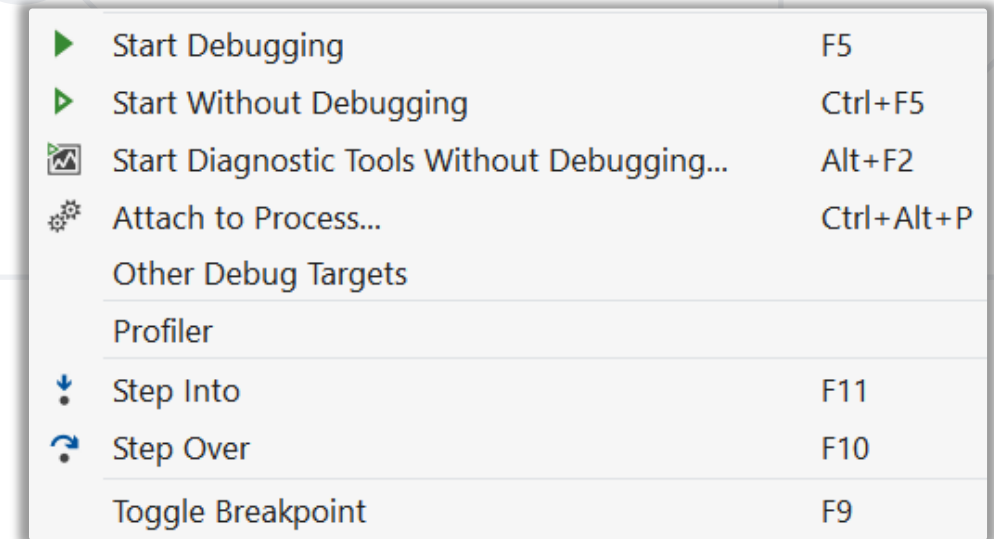
Debugging in Visual Studio

- Visual Studio has a built-in **debugger**
- It provides
 - **Breakpoints**
 - Ability to **trace** the code execution
 - Ability to **inspect** variables at runtime



Using the Debugger in Visual Studio

- Start without Debugger: **[Ctrl+F5]**
- Toggle a breakpoint: **[F9]**
- Start with the Debugger: **[F5]**
- Trace the program: **[F10]** / **[F11]**
- Using the **Locals** / **Watches**
- Conditional breakpoints
- Enter debug mode after exception



Problem: Find and Fix the Bugs in the Code

- A program aims to count the non-working days between two dates (e.g. 1.05.2016 ... 15.05.2016 → 5 non-working days). Debug it!

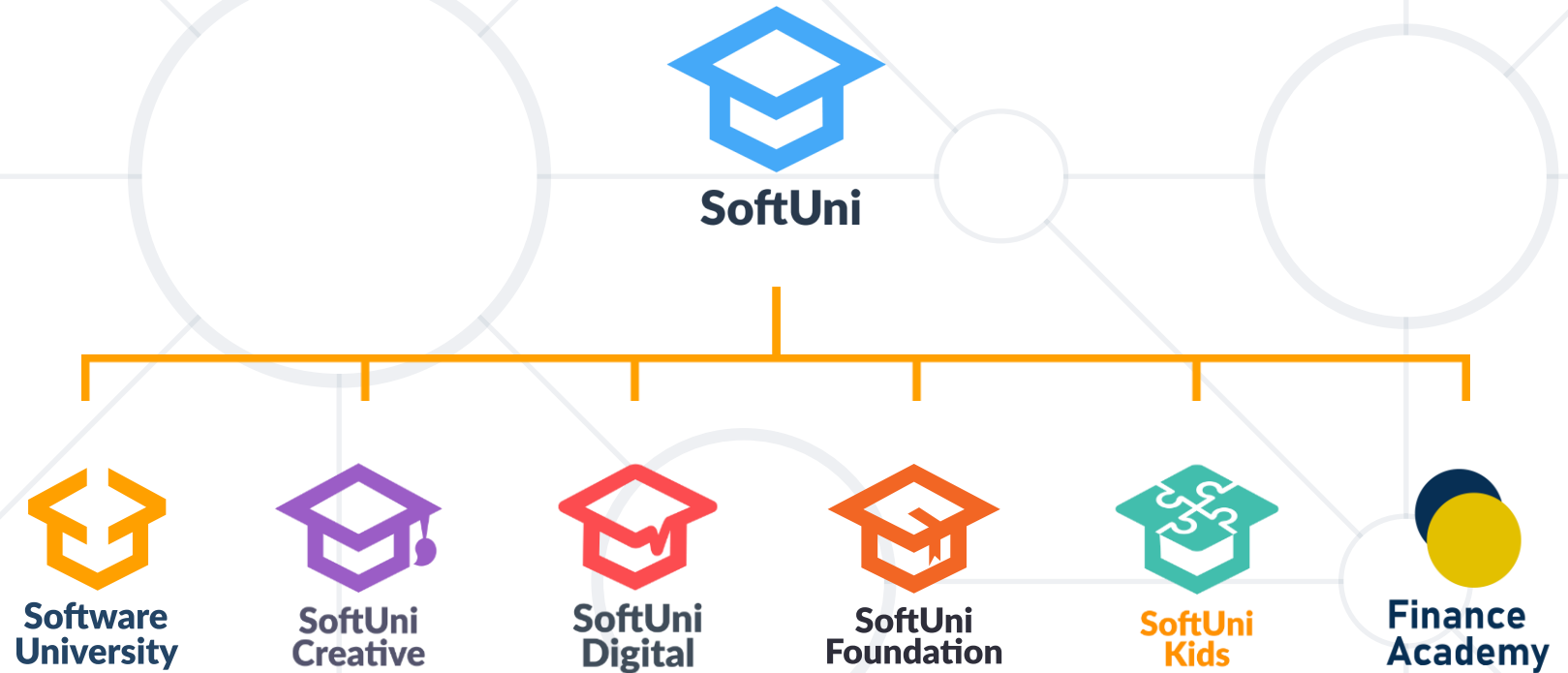
```
var startDate = DateTime.ParseExact(Console.ReadLine(),  
    "dd.m.yyyy", CultureInfo.InvariantCulture);  
var endDate = DateTime.ParseExact(Console.ReadLine(),  
    "dd.m.yyyy", CultureInfo.InvariantCulture);  
var holidaysCount = 0;  
for (var date = startDate; date <= endDate; date.AddDays(1))  
    if (date.DayOfWeek == DayOfWeek.Saturday &&  
        date.DayOfWeek == DayOfWeek.Sunday) holidaysCount++;  
Console.WriteLine(holidaysCount);
```

Check your solution here: <https://judge.softuni.org/Contests/Practice/Index/1188#12>

- Declaring **Variables**
- Using **Console – Reading** and **Writing**
- **Conditional Statements** allow implementing programming logic
- **Loops** repeat code block multiple times
- Using the debugger



Questions?



SoftUni Diamond Partners

**SUPER
HOSTING
.BG**



**Coca-Cola HBC
Bulgaria**



POKERSTARS
POKER | CASINO | SPORTS
a Flutter International brand

INDEAVR
Serving the high achievers



AMBITIONED


**DRAFT
KINGS**



**SOFTWARE
GROUP**

createX



Postbank
Решения за твоето утре

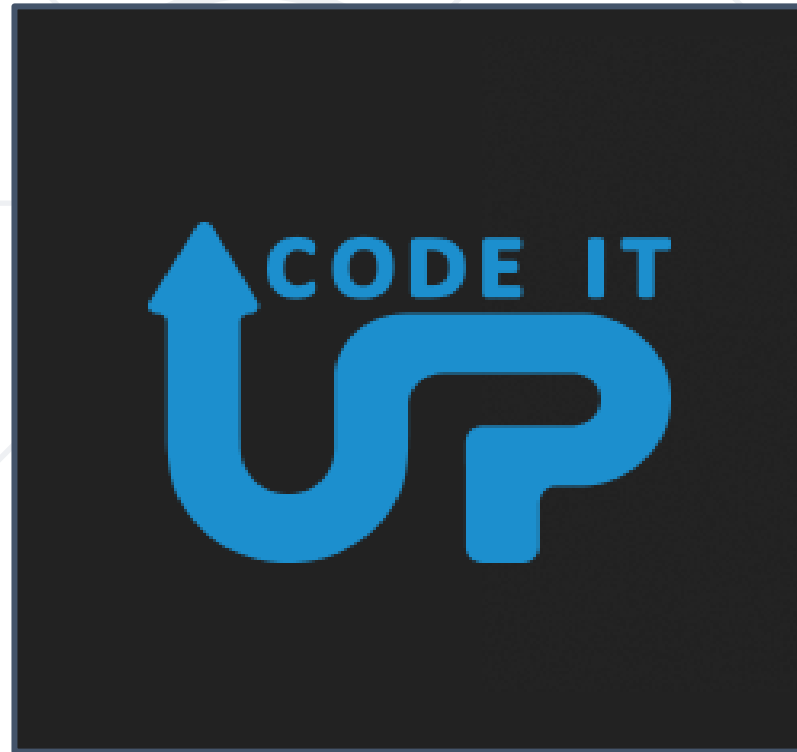


BOSCH

DXC
TECHNOLOGY



SmartIT



- Software University – High-Quality Education, Profession and Job for Software Developers

- softuni.bg, about.softuni.bg

- Software University Foundation

- softuni.foundation

- Software University @ Facebook

- facebook.com/SoftwareUniversity

- Software University Forums

- forum.softuni.bg



- This course (slides, examples, demos, exercises, homework, documents, videos and other assets) is **copyrighted content**
- Unauthorized copy, reproduction or use is illegal
- © SoftUni – <https://about.softuni.bg/>
- © Software University – <https://softuni.bg>

