

Дебъгване и оправяне на кода

Ред на изпълнение, дебъгване, препоръки
при писане на код



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>



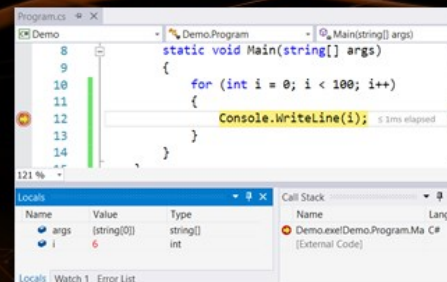
Съдържание



```
static void PrintHyphens(int count)
{
    Console.WriteLine(
        new string('-', count));
}

static void Main()
{
    for (int i = 1; i <= 10; i++)
    {
        PrintHyphens(i);
    }
}
```

Ред на изпълнение



Дебъгване на кода

Използване на дебъгера на Visual Studio



**Naming
Methods**

Методи

Именоване и добри практики

```
static void PrintHyphens(int count) ←  
{  
    Console.WriteLine(  
        new string('-', count));  
}  
  
static void Main()  
{  
    for (int i = 1; i <= 10; i++)  
    {  
        PrintHyphens(i);  
    }  
}
```

Ред на изпълнение

Изпълнение на програмата


- Програмата продължава след завършването на метода:

```
static void Main()  
{  
    Console.WriteLine("before method executes");  
    PrintLogo();  
    Console.WriteLine("after method executes");  
}
```

Първо е това

После е метода

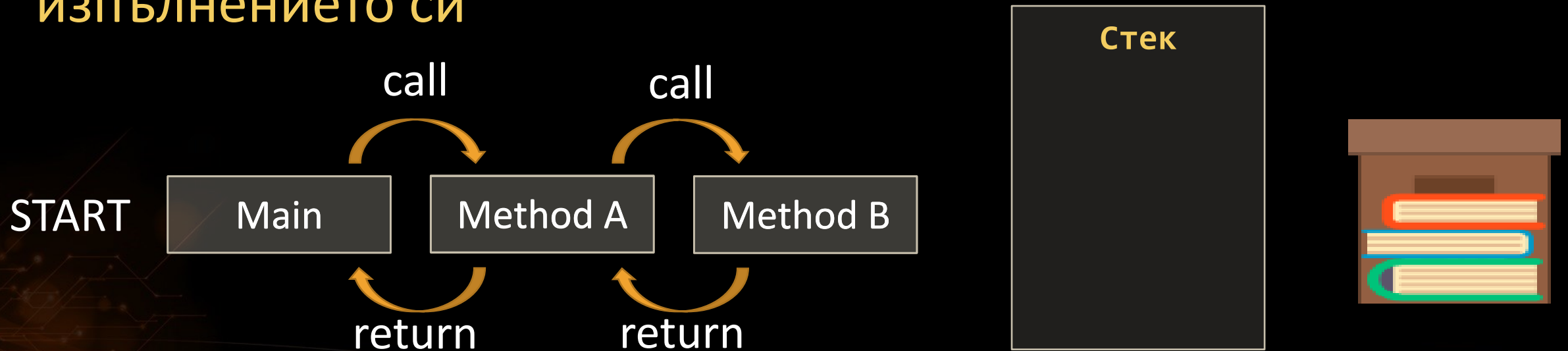
Накрая е това



```
static void PrintLogo()  
{  
    Console.WriteLine("Company Logo");  
    Console.WriteLine("http://www.companywebsite.com");  
}
```

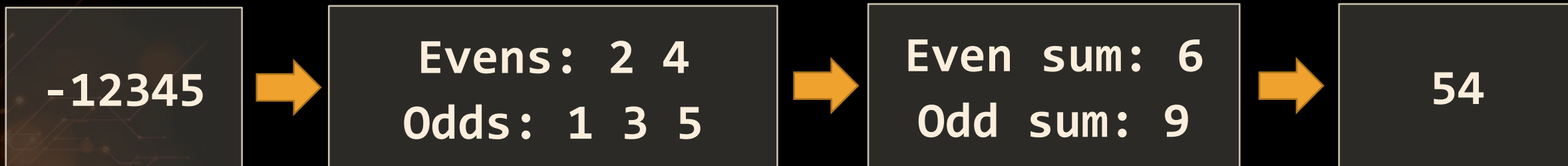
Ред на изпълнение – стек на извикванията

- „Стекът“ съдържа информация за активните подпрограми (методи) на текущо изпълняваната компютърна програма
- Пази информация за **точките**, към които всяка активна подпрограма трябва да **върне контрола**, когато тя **завърши изпълнението си**

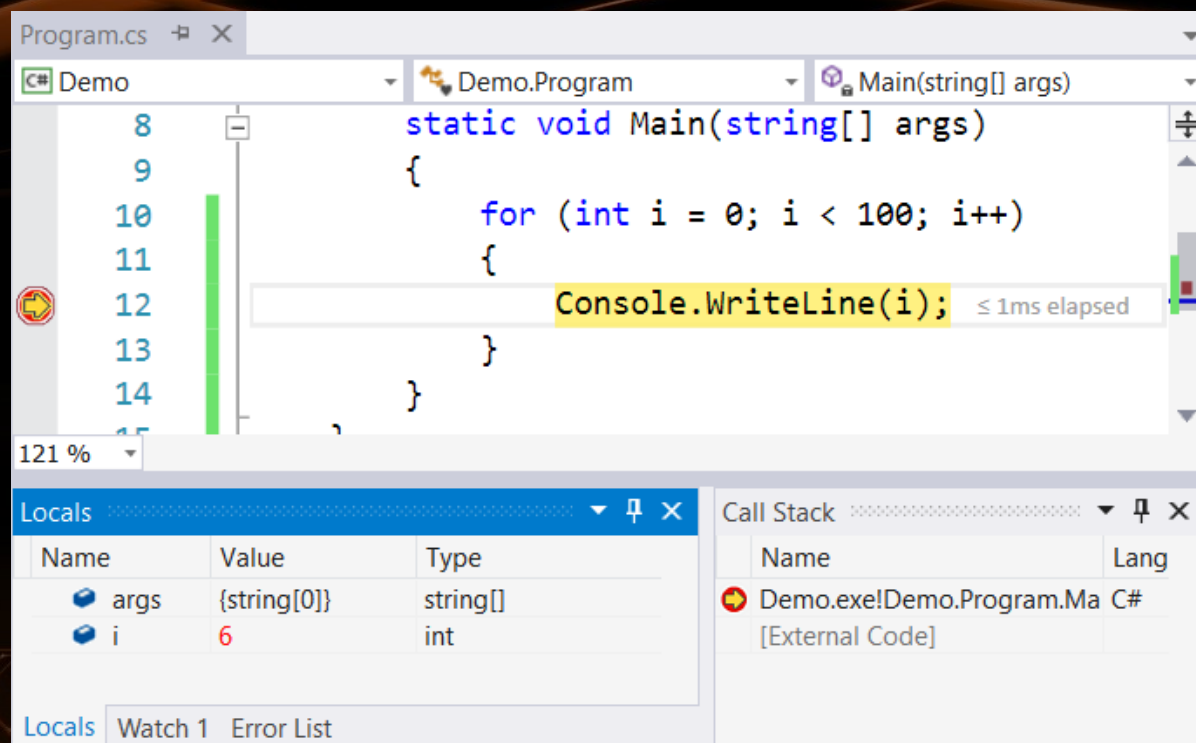


Задача: Умножи четни и нечетни цифри

- Създайте програма, умножаваща сумата на нечетните цифри на дадено число по сумата на четните му цифри:
 - Създайте метод наречен **GetMultipleOfEvensAndOdds()**
 - Създайте метод **GetSumOfEvenDigits()**
 - Създайте **GetSumOfOddDigits()**
 - Може да ви потрябва **Math.Abs()** за отрицателните числа



Тествайте в Judge: <https://judge.softuni.bg/Contests/2663>

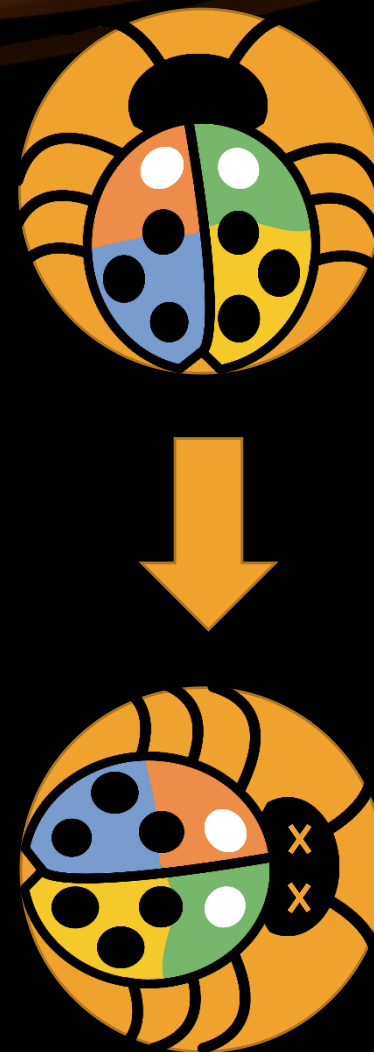


Дебъгване на кода

Използване на дебъгера на Visual Studio

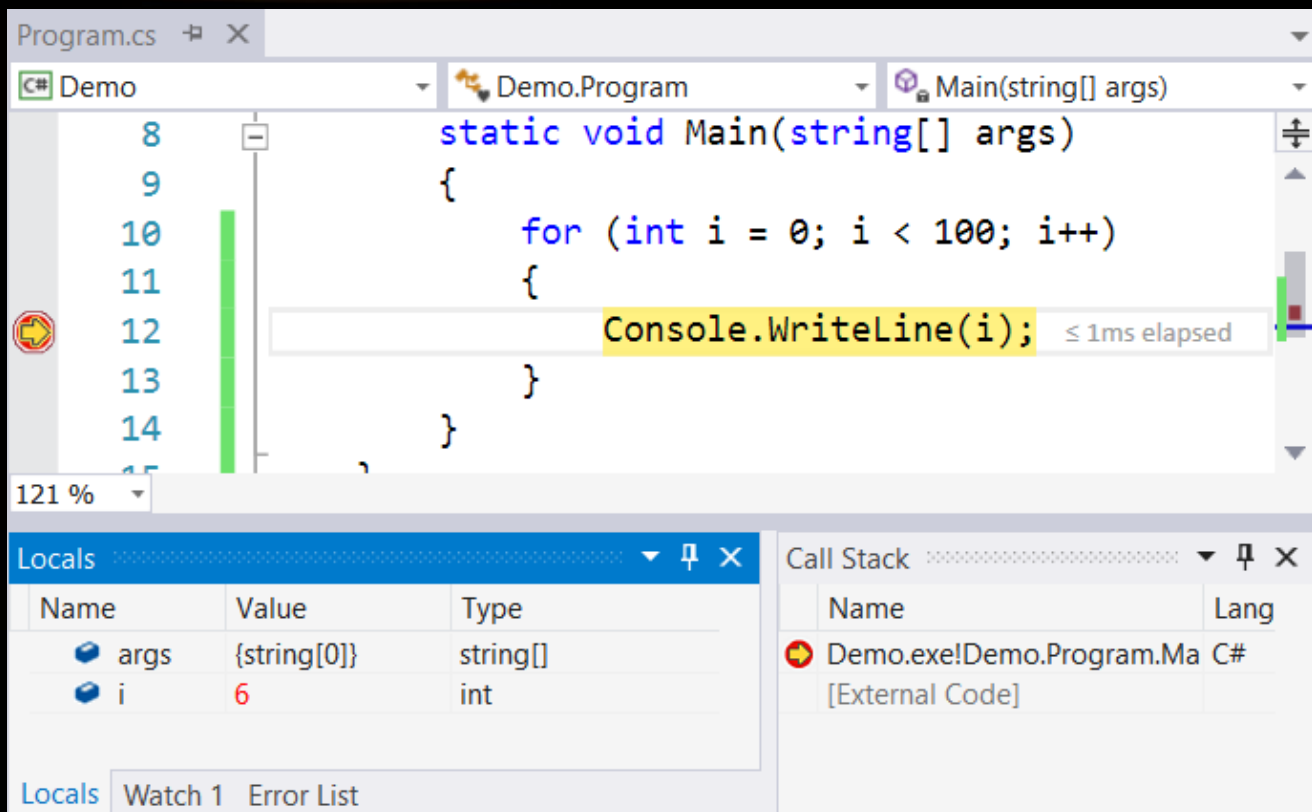
Дебъгване на кода

- Процесът на **дебъгване на програма** включва:
 - Откриване на грешка
 - Откриване на редовете в кода, които я предизвикват
 - Коригиране на грешката в кода
 - Проверка дали грешката е отстранена и дали междувременно не са добавени нови грешки
- Това е многократен и продължителен процес
- **Дебъгерът** помага много. Наистина помага!



Дебъгване във Visual Studio

- Visual Studio има вграден дебъгер
- Той ни предлага:
 - Стопери (breakpoints)
 - Възможност да **следим** изпълнението на кода
 - Средство за **наблюдение** на променливите по време на изпълнението на програмата



Използване на дебъгера във Visual Studio

- Стартиране без дебъгер: [Ctrl+F5]
- Активиране на стопер: [F9]
- Стартиране с дебъгер: [F5]
- Проследяване на кода: [F10] / [F11]
- Използване на Locals / Watches
- Условни стопери
- Дебъг режим след изключение

▶ Start Debugging	F5
▶ Start Without Debugging	Ctrl+F5
▶ Start Diagnostic Tools Without Debugging...	Alt+F2
⚙ Attach to Process...	Ctrl+Alt+P
Other Debug Targets	
Profiler	
⬇ Step Into	F11
⬇ Step Over	F10
⚙ Toggle Breakpoint	F9

Locals		
Name	Value	Type
▶ startDate	{01-Jan-15 00:00:00}	System.DateTime
▶ endDate	{02-Feb-16 00:00:00}	System.DateTime
▶ holidaysCount	2	int
▲ date	{10-Jan-15 00:00:00}	System.DateTime
▶ Date	{10-Jan-15 00:00:00}	System.DateTime
Day	10	int
DayOfWeek	Saturday	System.DayOfWeek
DayOfYear	10	int
Locals	Watch 1	

Задача: Намерете и поправете грешките

- Програмата се опитва да преброи **неработните дни между две дати** (напр. **1.05.2016 ... 15.05.2016** → **5** почивни дни). Дебъгнете я!

```
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);
```



Naming
Methods

Методи

Именоване и добри практики

Именуване на методи

- Препоръки при именуването на методи
 - Използвайте **описателни** имена на методи
 - Името трябва да отговаря на въпроса:
 - Какво прави този метод?



FindStudent, LoadReport, Sine

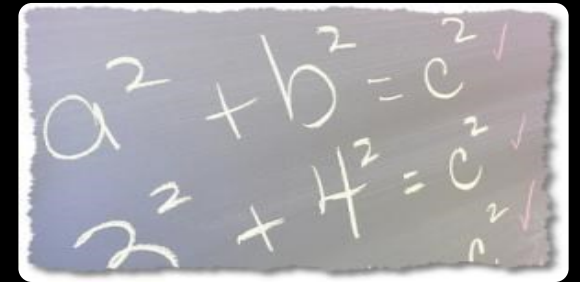
- Ако не намирате добро име за вашия метод, помислете дали той е с **ясно дефинирано предназначение**



Method1, DoSomething, HandleStuff, SampleMethod, DirtyHack

Именуване на параметрите на метод

- Имена на параметрите на метод
 - Препоръка: [Съществително] или [Прилагателно] + [Съществ.]
 - Трябва да е в **camelCase**
 - Трябва да е **говорящо**
 - Мерните единици трябва да са очевидни



`firstName, report, speedKmH,
usersList, fontSizeInPixels, font`



`p, p1, p2, populate, LastName, last_name, convertImage`

Добри практики при писане на методи

- Методът трябва да изпълнява **една** добре дефинирана задача
 - Името му трябва ясно и недвусмислено да **описва тази задача**
- Избягвайте методи, **по-дълги от един екран**
 - Разделете ги на няколко по-кратки метода


```
private static void PrintReceipt()  
{  
    PrintHeader();  
    PrintBody();  
    PrintFooter();  
}
```

Името обяснява всичко
и е лесно за тестване


Структура и форматиране на кода

- Подсигурете се, че коректно **вмъквате кода**

```
static void Main()  
{  
    ➡ // some code..  
    ➡ // some more code..  
}
```



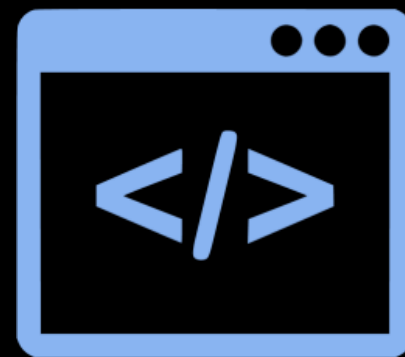
```
static void Main()  
    ➡ {  
        ➡ // some code..  
➡ // some more code..  
}
```



- Оставайте **празен ред** между **методите**, след **цикли** и след **if-команди**
- Тялото на цикли и if-команди ограждайте с **къдрави скоби**
- Избягвайте **дълги редове** и **сложни изрази**

Задача: Преправяване на "Price Change Alert"

- Програма, следяща **цени на стоки** и даваща информация за **значимостта** на всяка промяна в цената.
 - Изтеглете **програмния код** и се запознайте с него: **Broken-Solutions-1**
 - Дайте **подходящи имена на методите**
 - Поправете **имената на параметрите**
 - Погрижете се за **форматиране на кода**



`Get(c, 1)`



`GetPercentageDifference(currentPrice, lastPrice)`

Какво научихме този час?

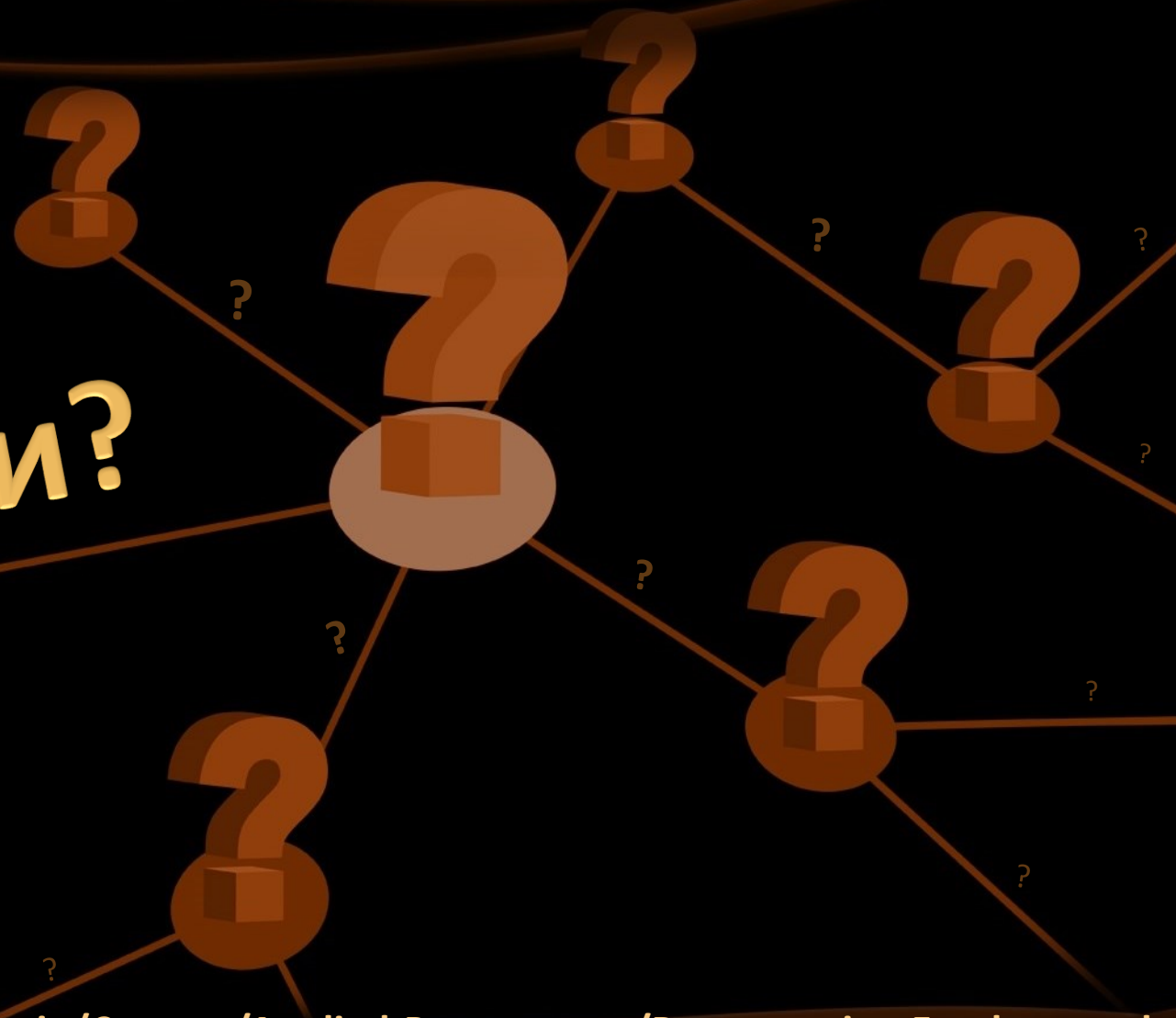
- Изпълнението на програмата продължава, когато **извиканият метод завърши**
 - **Стекът** съдържа активните подпрограми
- С **дебъгерът** следим изпълнението:
 - **стопери, постъпково изпълнение и наблюдение** на променливите
- Методите трябва да имат **една ясна цел**
- **Описателни имена** на методи и параметри
- Доброто **форматиране** на кода е важно



Дебъгване и оправяне на кода



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

