

Методи

Деклариране и извикване на методи



СофтУни

Трейнерски екип

Софтуерен университет

<http://softuni.bg>



Увод в
програмирането



1. Използване на методи

- Какво е метод?
- Защо използваме методи?
- Деклариране на методи
- Извикване на методи

2. Методи с параметри

- Използване на параметри в методите





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

Дефиниране и извикване на методи

Прости методи

- Метод е именована част от кода, която може да бъде извикана
- Примерна дефиниция на метод:

```
static void PrintHeader()
```

```
{
```

```
    Console.WriteLine("-----");
```

```
}
```

Метод, наречен
PrintHeader

Тялото на метода се
огражда с **{ }**

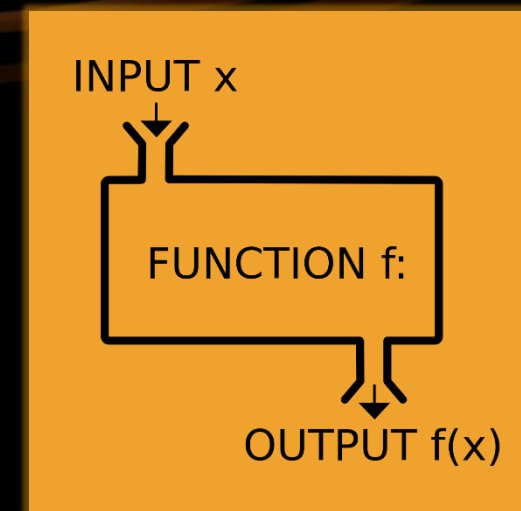
- Извикване на метода няколко пъти поред:

```
PrintHeader();
```

```
PrintHeader();
```


Защо да използваме методи?

- Програмирането става **по-обозримо**
 - Разделяме големите задачи на малки части
 - По-оптимална организация на програмата
 - Подобрява се четимостта на кода
 - Улеснява разбирането на кода
- Избягват се **повторенията в кода**
 - Улеснява поддръжката на кода
- **Повторно използване** на код
 - Използваме методите няколко пъти



Дефиниране на методи

тип на връщания
резултат

Име на метода

Параметри

```
static double GetSquare(double num)
```

```
{  
    return num * num;  
}
```

Тяло на
метода

- Методите се дефинират в **класа**
- **Main()** също е метод
- Променливите в метода са **локални**

```
class Program  
{  
    static void Main()  
    {  
    }  
}
```

Извикване на метод

- Методите първо се **дефинират**, а после **извикват** (многократно)

```
static void PrintHeader()  
{  
    Console.WriteLine("-----");  
}
```

Дефиниране
на метода

- Методите могат да бъдат **извикани** чрез името им + **()**:

```
static void Main()  
{  
    PrintHeader();  
}
```

Извикване
на метода

Извикване на метод (2)

- Метод може да бъде извикан от:

- Метода Main – **Main()**

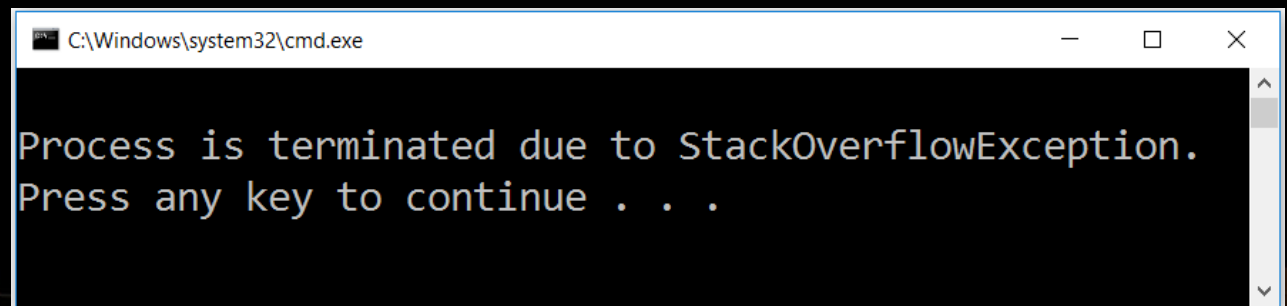
```
static void Main()  
{  
    PrintHeader();  
}
```

- Някой **друг метод**

```
static void PrintHeader()  
{  
    PrintHeaderTop();  
    PrintHeaderBottom();  
}
```

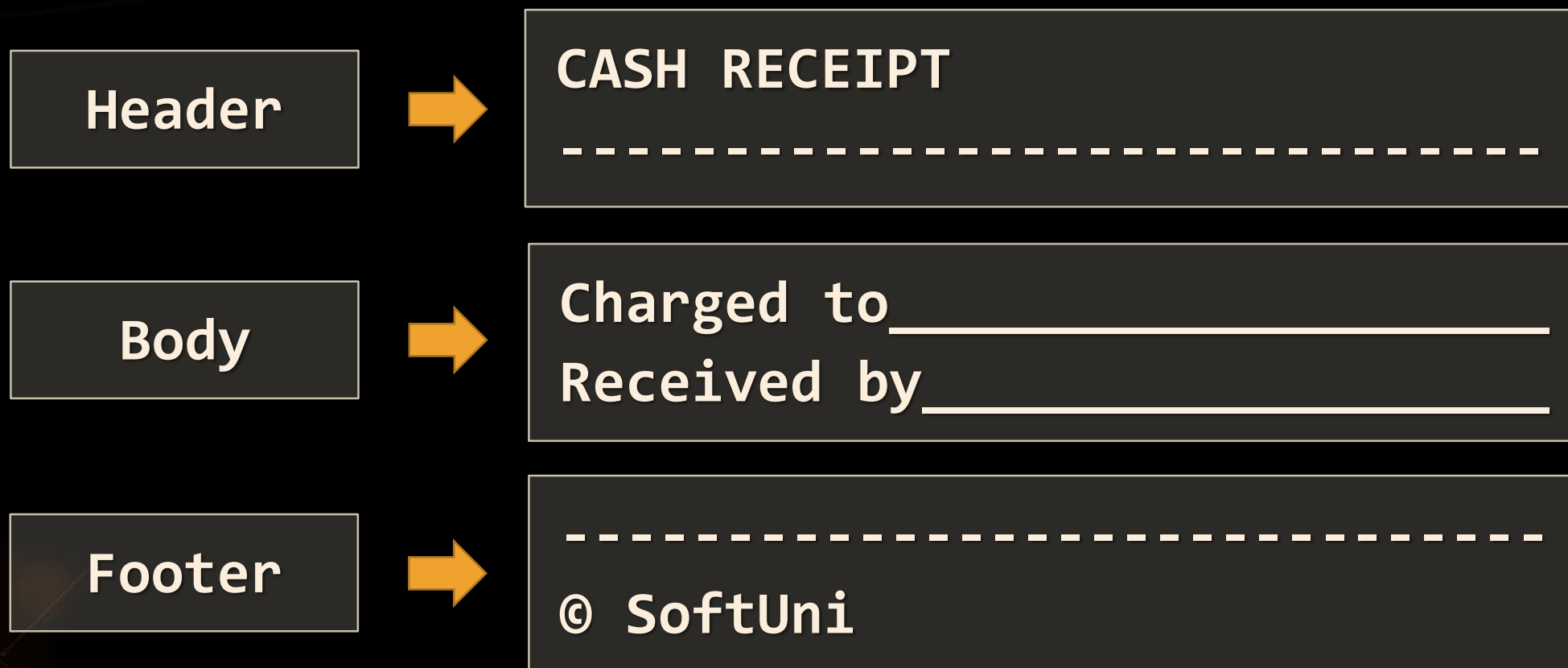
- **Своето тяло** – рекурсия

```
static void Crash()  
{ Crash(); }
```



Задача: Празна касова бележка

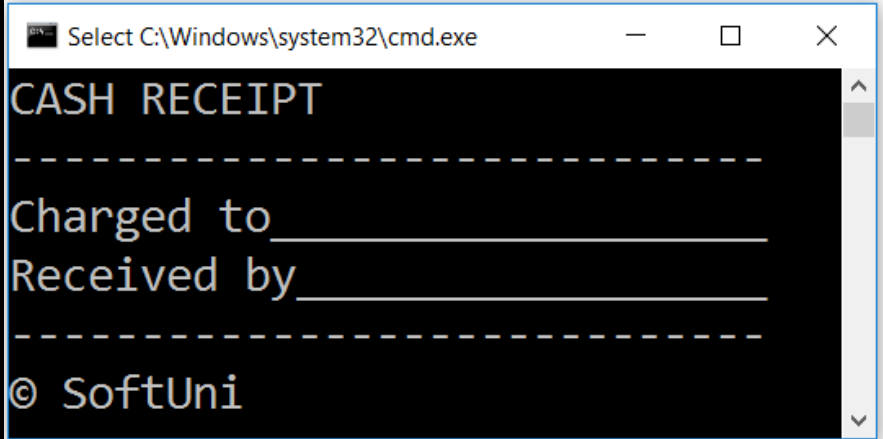
- Създайте метод, който отпечатва празна касова бележка:



Решение: Празна касова бележка

- Създайте **3 метода** за печат на секциите (header + body + footer)
 - Копирайте съдържанието от предния слайд
 - За символа © използвайте **Unicode** знака "**\u00A9**"
- Създайте метод **PrintReceipt()**, извикващ тези 3 метода:

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

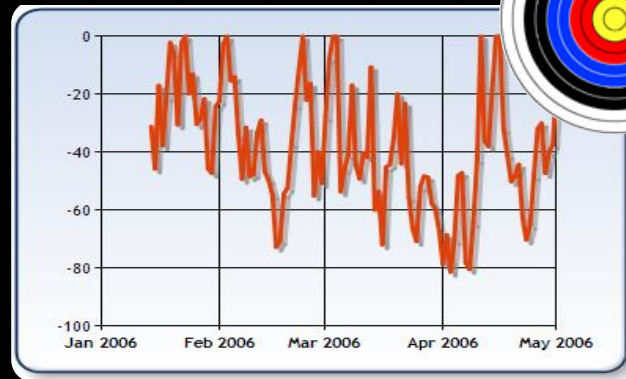


```
Select C:\Windows\system32\cmd.exe
CASH RECEIPT
-----
Charged to
Received by
-----
© SoftUni
```

int

long

double



float

decimal

Методи с параметри

Параметри на методите

- Параметрите могат да са от всеки тип данни

```
static void PrintNumbers(int start, int end)
{
    for (int i = start; i <= end; i++)
    {
        Console.Write("{0} ", i);
    }
}
```

Приема параметри
start и **end**
от тип **int**

Няколко параметъра,
разделени със запетая

- Извикването на метода е с конкретни стойности (**аргументи**)

```
static void Main()
{
    PrintNumbers(5, 10);
}
```

При извикване
подаваме аргументите

Параметри на методите (2)

- Може да подаваме **нула** или **повече** параметъра
- Параметрите могат да бъдат от **различен тип**
- Затова всеки параметър има **име** и **тип**

Няколко параметъра
от различен тип

Тип на
параметъра

Име на
параметъра

```
static void PrintStudent(string name, int age, double grade)
{
    Console.WriteLine("Student: {0}; Age: {1}, Grade: {2}",
        name, age, grade);
}
```

Задача: Знака на цяло число

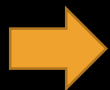
- Създайте метод, който отпечатва **знака** на цяло число **n**:

2



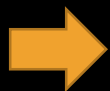
The number 2 is **positive**.

-5



The number -5 is **negative**.

0



The number 0 is **zero**.

Решение: Знак на цяло число

```
static void PrintSign(int number)
{
    if (number > 0)
        Console.WriteLine("The number {0} is positive", number);
    else if (number < 0)
        Console.WriteLine("The number {0} is negative.", number);
    else
        Console.WriteLine("The number {0} is zero.", number);
}

static void Main()
{ PrintSign(int.Parse(Console.ReadLine())); }
```

Опционални параметри

- Параметрите могат да имат стойности по подразбиране:

```
static void PrintNumbers(int start = 0, int end = 100)
{
    for (int i = start; i <= end; i++)
    {
        Console.Write("{0} ", i);
    }
}
```

Стойности по
подразбиране

- Методът по-горе може да бъде извикан по множество начини:

```
PrintNumbers(5, 10);
PrintNumbers(15);
PrintNumbers();
PrintNumbers(end: 40, start: 35);
```

Може да ги пропуснем при
извикването на метода

Задача: Отпечатване на триъгълник

- Създайте метод за отпечатване на триъгълници по начина, показан по-долу:

3



```
1
1 2
1 2 3
1 2
1
```

4



```
1
1 2
1 2 3
1 2 3 4
1 2 3
1 2
1
```

Решение: Отпечатване на триъгълник

- Създайте метод за печат на един ред от триъгълника, извеждащ числата от подаден **start** до подаден **end**:

```
static void PrintLine(int start, int end)
{
    for (int i = start; i <= end; i++)
    {
        Console.Write(i + " ");
    }
    Console.WriteLine();
}
```

Решение: Отпечатване на триъгълник (2)

- Създайте метод, печатащ първата част (1..n) и друг за втората част (n-1...1) от триъгълника:

```
static void PrintTriangle(int n)
{
    for (int line = 1; line <= n; line++)
        PrintLine(1, line);

    for (int line = n - 1; line >= 1; line--)
        PrintLine(1, line);
}
```

Метод с
параметър n

Редове 1...n

Редове n-1...1

Задача: Извеждане на запълнен квадрат

- Да се отпечати **запълнен квадрат** с размер **n** като в примера:

```
static void PrintHeaderRow(int n)
{
    Console.WriteLine(new
        string('-', 2 * n));
}
```

```
static void PrintMiddleRow(int n)
{
    Console.Write('-');
    for (int i = 1; i < n; i++)
        Console.Write("\\\\");
    Console.WriteLine('-');
}
```

Метод с
параметър **n**

4



```
-----
-\\\\-
-\\\\-
-----
```

```
static void Main() {
    int n = // TODO: read n
    PrintHeaderRow(n);
    for (int i = 0; i < n - 2; i++)
        PrintMiddleRow(i);
    PrintHeaderRow(n);
}
```


Какво научихме днес?

- Можем да разделим голяма програма на прости методи, които решават по-малки проблеми
- Методите имат
 - име, тип, параметри и тяло
- Методите се извикват по тяхното **име**
- Могат да приемат **параметри**
 - Параметрите приемат реални стойности, когато методът се извика



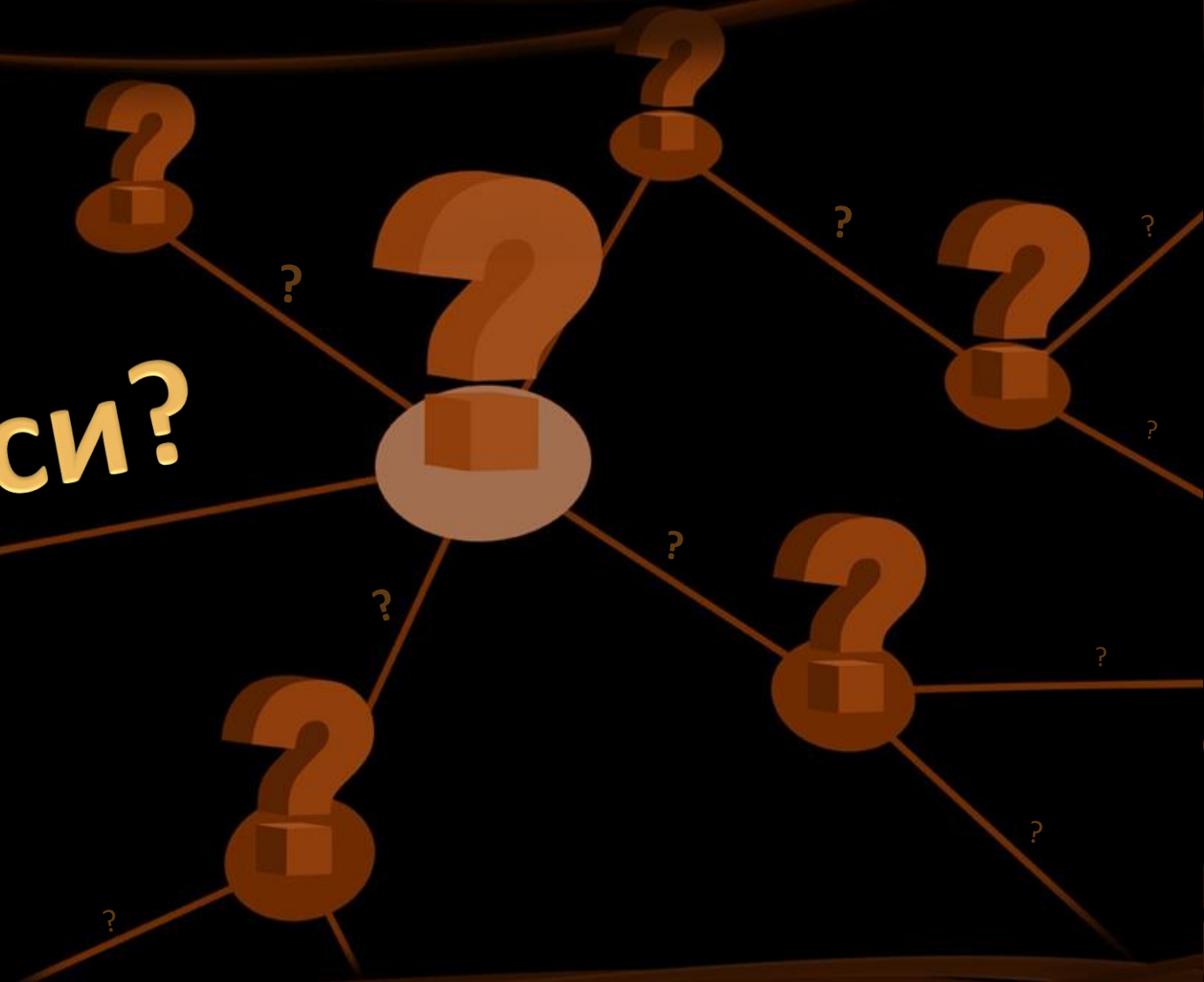
```
static void PrintHeaderRow(int n)
{
    Console.WriteLine(new
        string('-', 2 * n));
}

static void PrintMiddleRow(int n)
{
    Console.WriteLine('-');
    for (int i = 1; i < n; i++)
        Console.WriteLine("\\\\");
    Console.WriteLine('-');
}

static int SumOfDigits(int num)
{
    int sum = 0;
    while (num > 0)
    {
        sum += num % 10;
        num = num / 10;
    }
    return sum;
}
```



Въпроси?



- Настоящият курс (слайдове, примери, видео, задачи и др.) се разпространяват под свободен лиценз "Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International"



- Благодарности: настоящият материал може да съдържа части от следните източници
 - Книга "Основи на програмирането със C#" от Светлин Наков и колектив с лиценз CC-BY-SA