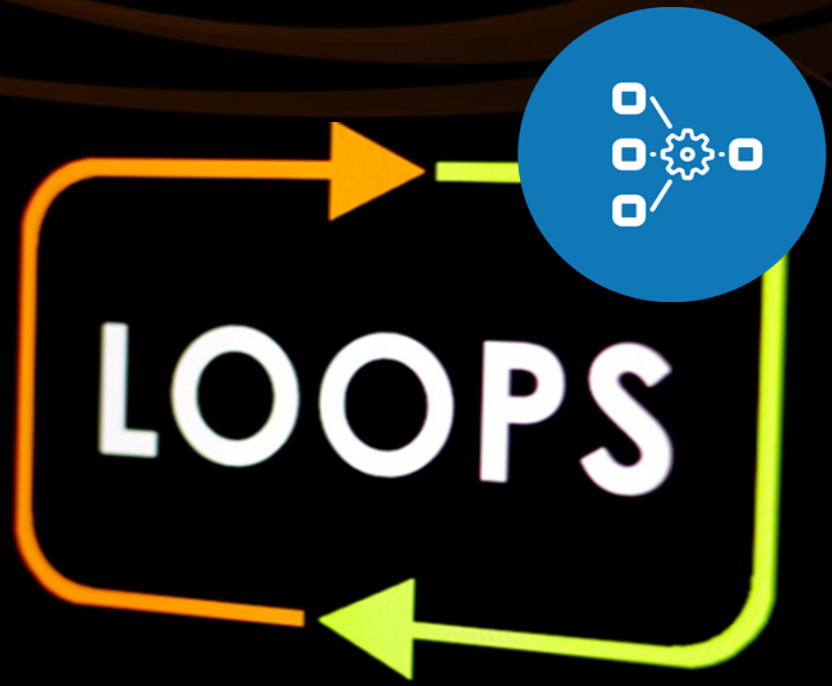
Повторения от по-висока сложност

Do…While, безкрайни повторения

Учителски екип

Обучение за ИТ кариера

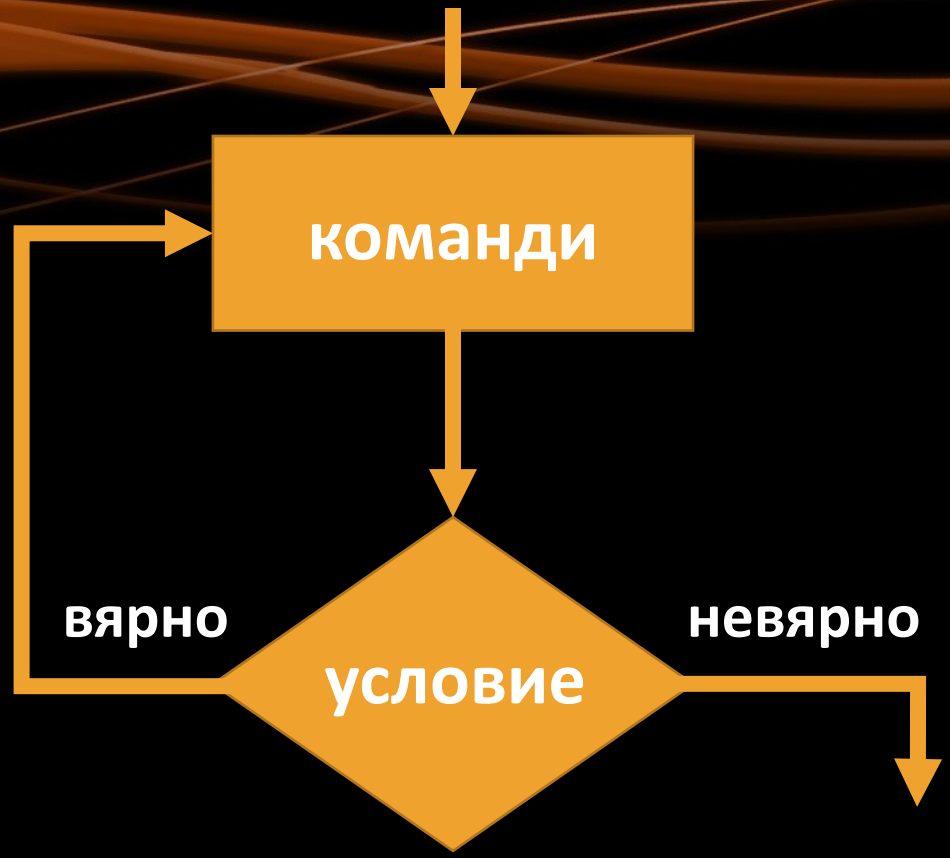
https://it-kariera.mon.bg/e-learning/



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

Съдържание

- По-сложни конструкции за цикъл:
  - `do-while` цикъл
  - безкраен цикъл
  - оператор `break`
  - `try…catch`
- По-сложни задачи с вложени цикли



Do…While цикъл

Повторение докато е изпълнено условието

Изчисляване на факториел

- За естествено число **n** да се изчисли **n! = 1 * 2 * 3 * … * n**
  - Пример: **5! = 1 * 2 * 3 * 4 * 5 = 120**

```
var n = int.Parse(Console.ReadLine());
var fact = 1;
do
{
    fact = fact * n;
    n--;
} while (n > 1);
Console.WriteLine(fact);
```

Тествайте в Judge:

https://judge.softuni.bg/Contests/2641/Повторения-от-по-висока-сложност

Сумиране на цифрите на число

- Да се сумират цифрите на цяло положително число **n**
  - При **n** = 5634: 5 + 6 + 3 + 4 = 18

```
var n = int.Parse(Console.ReadLine());
var sum = 0;
do
{
    sum = sum + (n % 10);
    n = n / 10;
} while (n > 0);
Console.WriteLine("Sum of digits: {0}", sum);
```

n % 10 връща последната цифра на числото n

n / 10 изтрива последната цифра на n

Тествайте в Judge:

https://judge.softuni.bg/Contests/2641/Повторения-от-по-висока-сложност

Безкрайни цикли и оператор Break

Безкраен цикъл

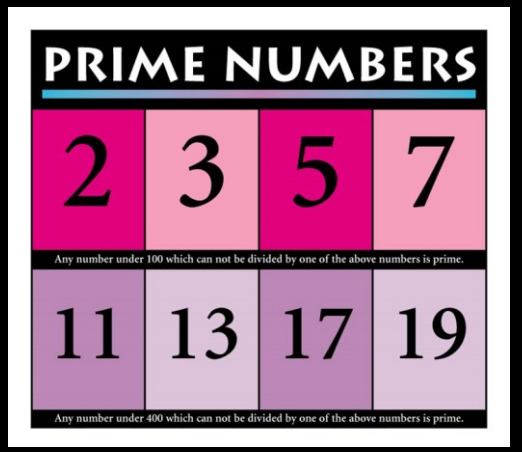
- Безкраен цикъл е когато повтаряме нещо до безкрайност:

```
while(true)
{
    Console.WriteLine("Infinite loop");
}
```

```
for (;;)
{
    Console.WriteLine("Infinite loop");
}
```

Прости числа

- Едно число **n** е просто, ако се дели единствено на **1** и **n**
  - Прости числа: 2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, …
  - Непрости (композитни) числа: 10 = 2 * 5, 21 = 3 * 7, 143 = 13 * 11
- Едно число **n** е просто, ако се дели на число между **2** и **n-1**
- Алгоритъм за проверка дали число е просто:
  - Проверяваме дали **n** се дели на 2, 3, …, n-1
  - Ако се раздели, значи е композитно
  - Ако не се раздели, значи е просто



- Оптимизация: вместо до n-1 да се проверяват делители до $\sqrt{n}$

Проверка за просто число. Оператор break

```
var n = int.Parse(Console.ReadLine());
var prime = true;
for (var i = 2; i <= Math.Sqrt(n); i++)
    if (n % i == 0) {
        prime = false;
        break;
    }
if (prime) Console.WriteLine("Prime");
else Console.WriteLine("Not prime");
```

break излиза от цикъла

Тествайте в Judge:

https://judge.softuni.bg/Contests/2641/Повторения-от-по-висока-сложност

Оператор Break в безкраен цикъл

- Да се напише програма, която въвежда четно число
  - При невалидно число да връща към повторно въвеждане

```
var n = 0;
while (true)
{
    Console.Write("Enter even number: ");
    n = int.Parse(Console.ReadLine());
    if (n % 2 == 0)
        break; // even number -> exit from the loop
    Console.WriteLine("The number is not even.");
}
Console.WriteLine("Even number entered: {0}", n);
```

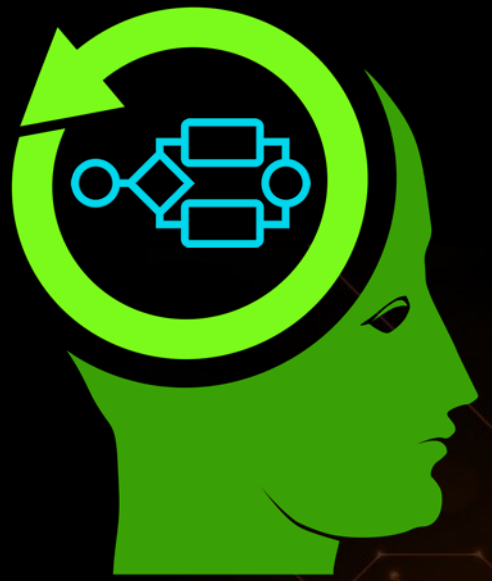
Тествайте в Judge:

https://judge.softuni.bg/Contests/2641/Повторения-от-по-висока-сложност

Справяне с грешни числа: try … catch

```
try
{
    Console.Write("Enter even number: ");
    n = int.Parse(Console.ReadLine());
    if (n % 2 == 0)
        break;
    Console.WriteLine("The number is not even.");
}
catch
{
    Console.WriteLine("Invalid number.");
}
```

Ако `int.Parse(…)` гръмне, ще се изпълни `catch { … }` блокът



Задачи с цикли

Числа на Фибоначи

- Числата на Фибоначи са следните: 1, 1, 2, 3, 5, 8, 13, 21, 34, …
  - $F_0 = 1$
  - $F_1 = 1$
  - $F_n = F_{n-1} + F_{n-2}$
- Пример: F(15) = 987
- Да се въведе **n** и да се пресметна **n**-тото число на Фибоначи

```
var n = int.Parse(Console.ReadLine());
var f0 = 1;
var f1 = 1;
for (var i = 0; i < n-1; i++)
{
    var fNext = f0 + f1;
    f0 = f1;
    f1 = fNext;
}
Console.WriteLine(f1);
```

Тествайте в Judge:

https://judge.softuni.bg/Contests/2641/Повторения-от-по-висока-сложност

Пирамида от числа

- Да се отпечатат числата 1…**n** в пирамида като в примерите:

```
n = 7
```

```
1
2 3
4 5 6
7
```

```
n = 10
```

```
1
2 3
4 5 6
7 8 9 10
```

```
n = 12
```

```
1
2 3
4 5 6
7 8 9 10
11 12
```

```
n = 15
```

```
1
2 3
4 5 6
7 8 9 10
11 12 13 14 15
```

Пирамида от числа – решение

```
var n = int.Parse(Console.ReadLine());
var num = 1;
for (var row = 1; row <= n; row++)
{
    for (var col = 1; col <= row; col++)
    {
        if (col > 1) Console.Write(" ");
        Console.Write(num);
        num++;
        if (num > n) break;
    }
    Console.WriteLine();
    if (num > n) break;
}
```

```
1
2 3
4 5 6
7 8 9 10
11 12
```

Тествайте в Judge:

https://judge.softuni.bg/Contests/2641/Повторения-от-по-висока-сложност

Таблица с числа

- Да се отпечатат числата 1…**n** в таблица като в примерите:

```
n = 2
```

```
1 2
2 1
```

```
n = 3
```

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

```
n = 4
```

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

```
n = 5
```

```
1 2 3 4 5
2 3 4 5 4
3 4 5 4 3
4 5 4 3 2
5 4 3 2 1
```

Таблица с числа – решение

```
var n = int.Parse(Console.ReadLine());
for (int row = 0; row < n; row++)
{
    for (int col = 0; col < n; col++)
    {
        var num = row + col + 1;
        if (num > n) num = 2 * n - num;
        Console.Write(num + " ");
    }
    Console.WriteLine();
}
```

```
1 2 3 4 5
2 3 4 5 4
3 4 5 4 3
4 5 4 3 2
5 4 3 2 1
```

Тествайте в Judge:

https://judge.softuni.bg/Contests/2641/Повторения-от-по-висока-сложност

Задачи с цикли

Работа на живо в клас (лаб)

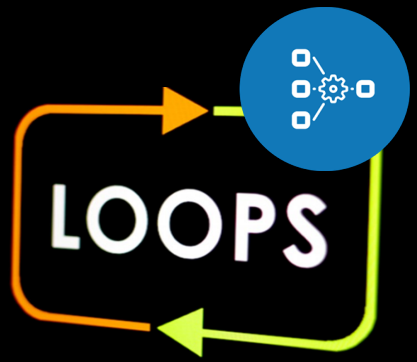
Какво научихме днес?

- Безкраен цикъл:

```
for (;;)
{ Console.WriteLine("Infinite loop"); }
```

- Прекъсване на безкраен цикъл
  - Оператор **break**

```
while(true)
{
   break;
   Console.WriteLine("This will not execute");
}
```



Повторения от по-висока сложност

Въпроси?

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

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"

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