



Увод в програмирането с Java



Цикли (I част)

Съдържание

- Какво е цикъл?
- Цикъл **while**
- Конструкция за цикъл **do-while**
- Безкрайни цикли
- Оператор **break**
- Оператор **continue**
- Задачи

Какво наричаме цикъл?

Конструкция за цикъл while

Задача

Да се изведат на екрана числата от 1 до 10.

Какво наричаме цикъл?

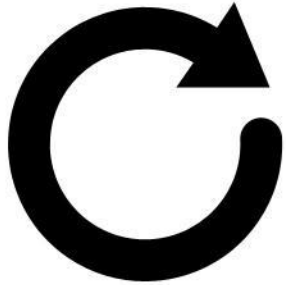
Понякога се налага да изпълняваме един и същи код многократно. За да не трябва повторно да пишем този код много пъти, в програмирането съществува концепцията за **цикъл (loop)** - повторено изпълнение на даден набор от операции. Всяко отделно изпълнение на операциите се нарича **итерация**. В Java съществуват три вида цикли - **for**, **while** и **do-while**.

Конструкция за цикъл `while`

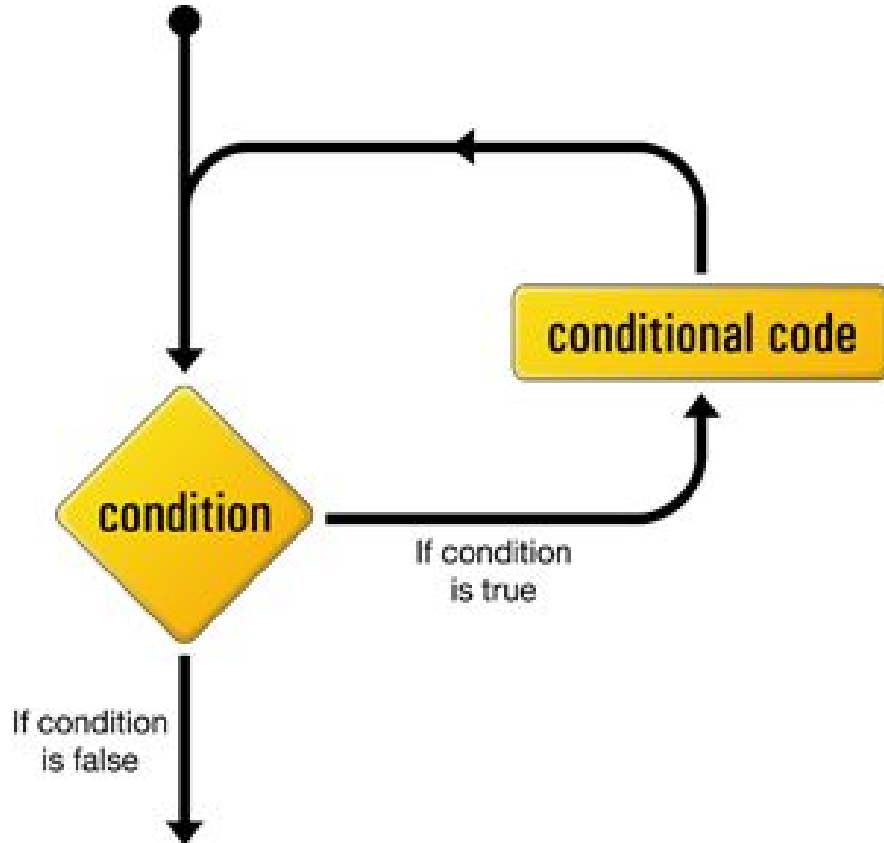
`while` изпълнява набор от операции, **докато** дадено условие е вярно:

```
while (условие) {  
    израз1;  
    израз2;  
    ...  
    изразn;  
    // код, който се изпълнява, ако условието е true  
}
```

Конструкция за цикъл while



LOOPS REPEAT
ACTIONS...
SO YOU DON'T HAVE TO



Задача

Да се изведат на екрана числата от 1 до 10, използвайки **конструкцията за цикъл *while***.

Задача

Да се напише програма, която пита потребителя да въвежда последователно в конзолата цели числа. Програмата да спира, когато потребителят въведе числото 0.

Задача

Да се напише програма, която намира сумата от цифрите на едно число.

Конструкция за цикъл do-while

Цикъл do-while

Конструкцията **do-while** е аналогична на **while**, като разликата е, че условието се оценява след изпълнението на операциите в цикъла (гарантираме **най-малко едно изпълнение**):

```
do {
```

```
statement1;
```

```
statement2;
```

```
...
```

```
}
```

```
while (boolean expression);
```

Цикъл do-while - пример

```
int number = 1;
```

```
do {
```

```
    System.out.println(number);
```

```
    number++;
```

```
} while (number <= 10);
```

Задача

Да се напише програма, която пита потребителя да въвежда последователно в конзолата цели числа. Програмата да спира, когато потребителят въведе числото 0.

Задача

Да се напише програма, която намира сумата на числата от 1 до n .

Задача

Да се напише програма, която намира произведението на цифрите на едно число.

Безкрайни цикли и специални оператори

Безкраен цикъл

Какво ще се случи при изпълнението на следния код?

```
while (true) {
```

```
System.out.println("Blablabla");
```

```
}
```

Безкраен цикъл

Безкрайният цикъл е цикъл, който никога не завършва. Създаването на безкраен цикъл трябва да се избягва, тъй като той кара програмата ни да “увисне” - да продължава да изпълнява едно и също нещо безкрайно, което от потребителска гледна точка не е желателно да става.

Оператор break

Операторът **break** се използва за прекъсване на цикъл. Всичко след него се пренебрегва и програмата излиза от цикъла:

```
while (условие) {  
  
    // код, който ще се изпълни  
  
    break;  
  
    // код, който няма да се изпълни  
  
}
```

Оператор break - пример

```
while (true) {  
    System.out.println("You'll see me");  
    break;  
    System.out.println("But you won't see me :(");  
}  
System.out.println("Out of loop");
```

Оператор break - пример

```
Scanner input = new Scanner(System.in);  
  
int number;  
  
while (true){  
  
    System.out.println("Please, insert a number: ");  
  
    number = input.nextInt();  
  
    if (number == 0)  
  
        break;  
  
}
```

Оператор `continue`

Операторът **`continue`** се използва за преминаване към следващата итерация. Всичко след него се пренебрегва, но програмата не излиза от цикъла, а се връща към оценяване на условието:

```
while (условие) {
```

```
// код, който ще се изпълни
```

```
continue;
```

```
// код, който няма да се изпълни,
```

```
а вместо него, ще се оцени наново условието
```

```
}
```


Оператор continue - задача

Да напишем програма, която извежда всички четни числа от 1 до n.

Задачи

Задача

1. Напишете програма, която изчислява факториела на числото n - произведението на числата от 1 до n .

Пример:

$$n! = 1.2.3.4...n$$

Задача

2. Напишете програма, която изчислява сбора на числата в интервала $[m, n]$.

Пример:

$$m = 2$$

$$n = 5$$

извеждаме: $2 + 3 + 4 + 5 = 14$

Задача

3. Напишете програма, която извежда всички числа, кратни на 2 и 3, в даден интервал $[m, n]$.

Пример:

Вход: $m = 10$

$n = 30$

Изход: 12 18 24 30

Задача

4. Напишете програма, която проверява дали едно число е просто. Просто число е число, което се дели само на 1 и на себе си.

Hint: Числото не трябва да се дели на нито едно от числата в интервала $[2, \sqrt{n}]$.

Задача

5. Напишете програма, която проверява дали едно число е “съвършено”. Едно число е съвършено, ако е равно на сумата от всичките си делители (без самото число, разбира се).

Задача - Остава за домашно

6. Напишете програма, която извежда едно число наобратно.

Задача - Остава за домашно

7. Напишете програма, която проверява дали едно число е палиндром. Палиндром е число, което четено отпред назад и отзад напред, е едно и също.

Задачи на листче :)

1. Каква е разликата между конструкциите за цикъл `while` и `do-while` и операторите `break` и `continue`?
2. Напишете програма, която извежда сумата на всички нечетни числа в интервала $[m, n]$ (m и n се въвеждат от конзолата).

Домашно

Конвенции, конвенции... :)

1. *Именуване на домашните - домашните се предават в zip/rar архив с име:*

HW5_LoopsPart1_<име на курсист>

2. *Архивът трябва да съдържа Java проект със същото име, като всяко решение на задача се намира в отделен клас (отделен .java файл)*

Неправилно именувани и/или подредени домашни няма да се зачитат!!

Задача

1. Напишете програма, която изчислява произведението на числата в интервала $[m, n]$.

Пример:

$$m = 2$$

$$n = 5$$

извеждаме: $2 \times 3 \times 4 \times 5 = 120$

Задача

2. Напишете играта “Отгатни числото”. Програмата кара компютъра да си намисли едно случайно число между 1 и 20, след което пита потребителя да въведе предположение. Ако предположението му не е вярно, програмата го моли да въведе ново число. Играта свършва, когато потребителят познае числото.

Hint: Как генерираме случайни числа?

```
Random rand = new Random();
```

```
int n = rand.nextInt(20) + 1;
```

Задача

3. Модифицирайте задача 2, като въведете брояч за опитите. След познаване на числото, програмата да изведе след колко опита е познато числото.

Задача

4. Модифицирайте задача 2, извеждайки насоки след всеки опит на потребителя - ако числото, което сме въвели, е по-голямо от намисленото, извеждаме “Too high, try again!”, ако пък е по-малко - “Too low, try again!”.

Задача

Прочетете точките за цикли, while и do-while от книгата (всичко до “Конструкция за цикъл for”):

[ЦЪК.](#)