PRINTF, SKRATENE RETURN VALUES, TERNARNY OPERATOR

PRINTF

Printf je iny sposob vypisovania niecoho do konzoly. Narozdiel od println() nezalomuje riadky automaticky, cize pre zalomovanie riadkov musime dotat znak \n.

```
System.out.println("hey");
System.out.printf("hey\n");
```
- tieto dva prikazy vypisu to iste do konzoly.

Vyhodou printf je, ze mozeme vystup viac formatovat. Ak chceme vypisat do konzoly nejaku premennu, musime uviest, akym formatom ju chceme napisat.
Napr.

%d – decimal (cele cislo)
%f – float (desatinne cislo)
%c – char (znak)
%s – string (retazec)

Kolko formatovacich % napiseme, tolko premennych musime uviest do argumentu metody printf.

Napr.

```
System.out.printf("Dnes je %d.%d.%d %d:%d.\n", 15,7,2023,12,51);
```
```
Dnes je 15.7.2023 12:51.
```

Ak chceme vypis formatovat lepsie, mozeme uviest viacero argumentov za znak %.
Ak namiesto %d napiseme %5d, zapiseme cislo na 5 miest. (alebo %5f – na 5 destinnych miest)

```
System.out.printf("Dnes je %d.%5d.%d %d:%d.\n", 15,7,2023,12,51);
```
```
Dnes je 15.    7.2023 12:51.
```

Ak napiseme %05d , tak cislo zapiseme na 5 miest. Miesta, kde nie su cisla sa zapisu nulami. (%05f – zapise sa cislo na 5 desatinnych miest)

```
System.out.printf("Dnes je %d.%05d.%d %d:%d.\n", 15,7,2023,12,51);
```
```
Dnes je 15.00007.2023 12:51.
```

Taktiez mozeme zapisovat stringy a znaky (treba rozlisovat “ a ‘):

```
System.out.printf("%s je %s. %c zajtra?\n", "Vonku", "horuco", 'A');
```
```
Vonku je horuco. A zajtra?
```

SKRATENY ZAPIS RETURNOV

Vacsinou sa vyuziva pre metody, ktore maju navratovu hodnotu typu boolean.

```java
private boolean jeStartMensi(int start, int end){
    if (start < end){
        return true;
    }else{
        return false;
    }
}
```

```java
private boolean jeStartMensi(int start, int end) {
    return start < end;
}
```

```java
public boolean jeDelitelne(int cislo, int delitel){
    if (cislo % delitel == 0){
        return true;
    }
    return false;
}
```

```java
public boolean jeDelitelne(int cislo, int delitel){
    return cislo % delitel == 0;
}
```

Metody vratia to iste.

Taktiez pre jednoriadkove if-else if-else podmienky sa nemusia pisat kucerave zatvorky (no nie je to chyba, ak sa kucerave zatvorky napisu):

```java
private boolean jeStartMensi(int start, int end){
    if (start < end)
        return true;
    else
        return false;
}
```

TERNARNY OPERATOR

Ternarny operator nie je nic ine, ako skratene napisana if podmienka. Je dobre pisat ternarny operator vtedy, ak if podmienka je velmi kratka.

Struktura if podmienky:
if (vyraz){
  toto sa stane, ak je vyraz true;
}else{
  toto sa stane, ak je vyraz false;
}
Struktura ternarneho operatora:
vyraz ? toto sa stane, ak je vyraz true : toto sa stane v opacnom pripade

```java
if (ify.jeKladne(cislo1))
    System.out.printf("Cislo %d je kladne\n", cislo1);
else
    System.out.printf("Cislo %d je zaporne\n", cislo1);
```

```java
System.out.printf("Cislo %d je %s\n",cislo1, ify.jeKladne(cislo1) ? "kladne" : "zaporne");
```

Vsetky tieto funkcie nemenia funkcionalitu programu, iba zlepsuju vzhlad kodu, aby bol lepsie citatelny.