14. Funkcije in parametri bolj podrobno

14.1. Privzete vrednosti formalnih parametrov.
Formalnim parametrom lahko predpišemo pri definiciji privzete vrednosti. Take parametre napišemo na koncu, za vsemi navadnimi parametri. Snov bomo bolj podrobno obravnavali na enem od kasnejših predavanj.

```
def f(x,y=1,z=0):
      A
```

Ko funkcijo `f` pokličemo
`f(a)`, je to v bistvu `x,y,z = a,1,0`.
`f(a,b)` je `x,y,z=a,b,0`
`f(a,b,c)` pa `x,y,z=a,b,c`.

14.2. Imenovani paramateri
Funkcijo lahko pokličemo, tako da formalnemu parametru priredimo vrednost dejanskega parametra.

```
def f(a,b):
....
```

Klic `f(b=7,a=2)`
(Pozor, pri takem načinu klicanja vrstni red parametrov pri klicu ni pomemben)

14.3. Zgled:
Želimo oblikovati šahovnico dimenzije n x n, pri čeme uporabimo znaka crno in belo za risanje črno-belih polj. Privzeti parametri naj bodo 8,"*" in " ".

```
def sahovnica(n=8,crno="*",belo=" "):
#vrnemo sahovnico n x n s polji barve crno
#in belo.
...
```

14.4. Funkcije s poljubnim številom parametrov

- Funkcijo lahko definiramo tako, da ji bomo pri klicu lahko podali poljubno število parametrov (neimenovanih ali imenovanih)
- **`def f(a, b,*s, c = 3, **t)`**
- **`s`** je nabor, ki bo po klicu funkcije vseboval vse dodatne neimenovane parametre.
- **`t`** je slovar, ki bo po klicu funkcije vseboval vse dodatne imenovane parametre. To sibomo pogledali, ko bomo spoznali slovarje.

14.5. Zgled. Denimo, da bi želeli izračunati dolžino vektorja, ki je dan s komponentami.

```
def dolzina(*v):
      s = 0
      for x in v:
            s += x*x
```

```
        return s**(1/2)
```

Na videz je funkcija podobna naslednji:

```
def dolzina1(v):
        s = 0
        for x in v:
                s += x*x
        return s**(1/2)
```

14.6. Funkcije kot parametri

- Tudi funkcija je lahko parameter drugi funkciji.
- Klasičen primer je metoda **`sort`**, ki zna urediti elemente seznama. Kot parameter z imenom **`key`** ji lahko podamo funkcijo, ki elementu seznama priredi njegovo vrednost. Metoda **`sort`** bo potem urejala elemente seznama glede na vrednosti, ki jih vrača funkcija.

14.7. Anonimne funkcije

- Če kakšno preprosto funkcijo potrebujemo samo na enem mestu, nam je ni treba poimenovati
- Anonimno funkcijo definiramo s pomočjo rezervirane besede **`lambda`**, za katero napišemo imena prametrov, dvopičje in izraz, ki določa, kako se izračuna vrednost funkcije
- **`lambda x: x ** 3`**
- **`lambda x, y: x + len(y) - 2`**

14.8. Naloge:

1. Sestavite funkcijo, ki izpiše vse svoje parametre (obvezne, privzete, dodatne nepoimenovane in dodatne poimenovane).

14.9. Funkcija print bolj podrobno

Kot že vemo, ima funkcija print lahko poljubno število parmetrov. Med njimi ima lahko še tri posebne imenovane parametre.

Parameter: `sep`
S parametrom sep nastavimo separator med posameznimi natisnjenimi izrazi. Privzeta vrednost je presledek `sep = " "`.

Parameter: `end`
S parametrom end uravnavamo dogajanje po koncu tiskanja.
Privzeta vrednost je `end = "\n"`, torej skok v novo vrsto.

Parameter: `file`

S parametrom file lahko pišemo na poljubno datoteko, ki je odprta za pisanje. Privzeta vrednost je standardni izhod `sys.stdout`.

Posebni znaki:

`"\n"` ... skok v novo vrsto
`"\t"` ... tabulator