16. Slovarji

Slovar je sestavljena podatkovna struktura, v kateri hranimo 'poimenovane' vrednosti (podobno, kot v seznamih hranimo indeksirane vrednosti). Vsaka vrednost ima svoje 'ime', ki mu rečemo ključ. Vrednosti se lahko podvajajo, ključi pa ne.

- Prazen slovar zapišemo s praznimi zavitimi oklepaji: **`{}`**
- Neprazen slovar zapišemo v zavitih oklepajih, v katerih naštejemo ključe in njihove vrednosti: **`{k1: v1, k2: v2, ..., kn: vn}`**
- Ključi morajo biti paroma različni.
- Ključi so lahko nizi, števila, terice (ki ne vsebujejo seznamov), ...
- Vrednosti so lahko poljubnih tipov.

16.1. Slovar je naravna posplošitev množice in seznama.

Slovar je novejša podatkovna struktura, ki je starejši programski jeziki ne poznajo in so jo morali nekoč programerji sprogramirati "peš". Najlaže si jo predstavljamo, če najprej pogledamo, kaj imata skupnega seznam in množica. Pri množicah je npr. pomembno, da se elementi ne morejo pojavljati večkrat, medtem, ko se v seznamih vrednosti lahko pojavijo večkrat. Res pa je, da so v seznamu indeksi enolični. V seznamih indeks igra vlogo ključa, pri množicah pa je kar element ključ. Do slovarja po premisleku pridemo tako po dveh poteh:
1. Lahko si mislimo, da vsakemu elementu množice (ključ) dodamo še vrednost.
2. Lahko si mislimo, da v seznamu pojem indeksa s števila razširimo na druge objekte. V tem drugem pristopu se moramo zavedati, da s posplošitvijo indeksov izgubimo linearno ureditev ključev, ki je bila pri naravnih številih očitna.
Slovar je torej množica, pri kateri imajo elementi vrednost, ali pa seznam, v katerem indeksi niso le zaporedna števila 0,1,... Ta dvojni pogled na slovarje se izraža tudi pri njihovi sintaksi: med praznim slovarjem in prazno množico ni razlike:{}. V izrazu s[k] pa ne moremo razločiti med seznamom in slovarjem.

16.2. Operacije nad slovarji:

- **`s[k]`** ... vrednost s ključem k v slovarju s (če ključa ni v slovarju, javi napako)
- **`s[k] = v`** ... nastavi vrednost, ki pripada ključu k v slovarju s
- **`del s[k]`** ... odstrani ključ k in pripadajočo vrednost iz slovarja s
- **`k in s`** ... ali je ključ k v slovarju s
- **`len(s)`** ... število ključev v slovarju s

16.3. Metode nad slovarji:

- **`s.clear()`** ... sprazni slovar s
- **`s.get(k)`** ... vrne vrednost ključa k v slovarju s (če ključa ni v slovarju, vrne None)
- **`s.get(k, v)`** ... vrne vrednost ključa k v slovarju s (če ključa ni v slovarju, vrne v)

16.4. Zanke po slovarju

- **`for k in s: ...`** zanka po vseh ključih slovarja s
- **`for k in s.keys(): ...`** zanka po vseh ključih slovarja s
- **`for v in s.values(): ...`** zanka po vseh vrednostih slovarja s
- **`for k, v in s.items(): ...`** zanka po vseh parih (ključ, vrednost) slovarja s

16.5. Zgled. Frekvence.

Sestavimo funkcijo, ki v danem nizu prešteje, kolikokrat se pojavi posamezen znak.

```
def frekvence(niz):
"""vrnemo slovar, tako da pomeni slovar[znak] število pojavitev znaka
znak v nizu niz."""
slovar = {}
for znak in niz:
      slovar[znak] = slovar.get(znak,0) + 1
return slovar
```

16.6. Zgled. Ključ najmanjšega elementa.

Napišimo funkcijo, ki za dani slovar izbere ključ, ki ima najmanjšo vrednost.

```
def minumum(slovar):
      """vrnemo kljuc, ki ima v slovarju slovar najmanjso vrednost.
      """
      if len(slovar) == 0:
            return None
      for k in slovar.keys():
            vmin = slovar[k]
            kmin = k
            break
      for k,v in slovar.items():
            if v < vmin:
                  kmin = k
                  vmin = v
      return kmin
```

16.7. Naloge

1. Sestavite funkcijo, ki za parameter dobi seznam imen, vrne pa nov seznam z istimi imeni, le da na koncu vsakega imena, ki se pojavi večkrat, doda zaporedno rimsko številko.
2. Sestavite funkcijo, ki omogoča osnovno prevajanje iz slovenščine v angleščino (prevajanje po besedah). Funkcija naj ohranja velike začetnice.
3. Sestavite funkcijo, ki iz seznama točk izdela slovar, katerega ključi so vse različne koordinate x, vrednosti pa ustrezni seznami koordinat y.