13. Rekurzija

Splošna ideja rekurzije:

(1. - REKURZIJA) problem se da reducirati na manjši problem enakega tipa,
(2. - ZAUSTAVITVENI POGOJ) dovolj majhen problem pa znamo rešiti neposredno.

Lahko bi rekli, da pri rekurzivnem razmišljanju idejo "kreativne lenobe" pripeljemo do skrajnosti, saj nam v rekurzivnem koraku problema ni potrebno rešiti, ampak ga reduciramo nase. Pozor: potreben je zaustavitveni pogoj, sicer se rekurzija nikoli ne konča.

Python omogoča, da funkcija pokliče samo sebe. Včasih lahko znane rekurzivne definicije neposredno prepišemo v funkcijo. Npr.

13.1 Zgled

0! = 1
n! = n (n-1)! za n > 0.

V Pythonu to zapišemo takole:

```
def fakulteta(n):
     if n == 0:
          return 1
     return n*fakulteta(n-1)
```

13.2 Značilnosti rekurzivne funkcije

- Funkcija lahko poleg drugih funkcij kliče tudi sama sebe, lahko tudi posredno.
- Paziti moramo, da se ne bo klicala neskončno mnogokrat.
- Na začetek običajno dodamo pogojni stavek, s katerim preprečimo nadaljne klice, ko to ni več potrebno.
- Ob nepazljivi uporabi so lahko rekurzivne funkcije zelo počasne.
- Pri funkcijskem programiranju so rekurzivne funkcije pogosto nepogrešljive.

13.3 Naloge:

1. Sestavite funkcijo `fakulteta(n)`, ki s pomočjo rekurzije izračuna n!. Pri tem uporabite formulo n!=n*(n-1)! ter 0! = 1! =1.
2. Sestavite funkcijo `obrni(niz)`, ki s pomočjo rekurzije obrne podani niz. Funkcija naj vrne obrnjen niz
3. Sestavite funkcijo `izpisiObrnjen(niz)`, ki s pomočjo rekurzije podani niz obrne in ga izpiše. Pazite na to, da ne izpisujete vsakega znaka v svojo vrstico (uporabite `end=''`, ko kličete `print`)
4. Sestavite funkcijo, ki s pomočjo rekurzije prešteje število razdelitev n elementov na k nepraznih podmnožic. Dobljeno število označimo z S(n,k). To so Stirlingova števila druge vrste. Predpostavimo, da je n >= k.
   Namig: ko računamo S(n+1,k), je bodisi {n+1} ena od podmnožic (v tem primeru

je preostalih n elementov v k-1 podmnožicah) bodisi n+1 ni sam zase (v tem primeru je preostalih n v k podmnožicah, za n+1 pa imamo potem k možnosti). Torej velja:
$S(n+1,k) = S(n,k-1) + k*S(n,k)$
Kaj pa zaustavitveni pogoj? Očitno velja $S(n,1)=1$ (vsi elementi so skupaj) in $S(n,n)=1$ (vsak element je zase).

5. Sestavite tabelo Stirlingovih števil druge vrste, nato pa izpišite * za vsako liho število v tabeli in presledek za vsako sodo. Kakšen vzorec opazite? (odvisno od tega, kako ste napisali funkcijo za Stirlingova števila, boste lahko vrednosti najverjetneje tabelirali samo za n=1,...,20 in k=1,...,n.)
6. Sestavite rekurzivno funkcijo, ki bo izračunala n-ti člen fibonaccijevega zaporedja.
7. Sestavite rekurzivno funkcijo, ki bo preverila, ali je dana beseda palindrom.
8. Sestavite rekurzivno funkcijo, ki bo za dani seznam generirala vse razpršene podsezname.
9. Sestavite rekurzivno funkcijo, ki bo za dano množico generirala seznam njenenih podmnožic. (Potenčna množica)
10. Uredite seznam imen tako, da bodo imena urejana najprej po njhovi dolžini, imena z isto dolžino pa še leksikografsko.