

# Iteratorji & generatorji

Razred  $C$ , objekti iz  $C$  so "zbirke" podatkov:

- tabela
- slovar ("podatki" = ključi)
- datoteka (podatki = vrstice)
- graf (vozlišča)

Iz objekta  $o : C$  naredimo iterator:

- objekt, ki ima
- metodo `--next--`, ki ob vsakem klicu vrne naslednji element zbirke  $o$   
(ko elementov zmanjka, sproti izjemo `StopIteration`)
- tehnična zahteva: metoda `--iter--`, ki vrne `self`.

Kako uporabimo iterator:

1. Iterator za objekt  $o$  dobimo z `iter(o)`
2. Če je  $i$  iterator z `next(i)` dobimo naslednji element.
3. V zanki `for`:

for x in y:  
B.

- Python pokliče `iter(y)`, kar je ekvivalentno `y.__iter__()`, da dobi iterator `it`
- na `it` kliče `next(it)`, ekvivalentno `it.__next__()`, in dobiva elemente, ki jih po vrsti daje v `x` in vsaki izvede B.

## Generatorji

"Funkcija", ki računa elemente/zaporedje za iterator. Namesto return uporabimo yield

- Vsaki, ko se izvede `yield`, dobimo naslednji element in izvajanje generatorja se ustavi
- Ob naslednjem klicu `next` se izvajanje nadaljuje.

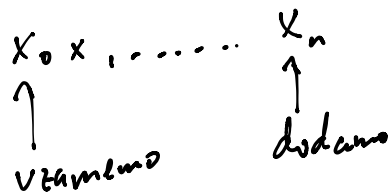
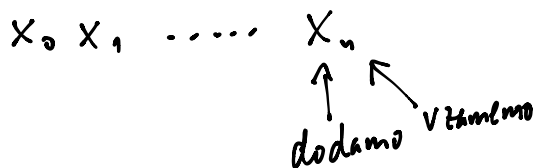
# Podatkovne strukture

- Kako v prog. jeziku organiziramo podatke
- Kaj bomo počeli s podatki? Od odgovora je odvisno, kako jih bomo organizirali.

## Primeri

### Shlad & vrsta

- zaporedje podatkov  $x_0, x_1, \dots, x_{n-1}$
- Sklad:
  1. Naredi prazen sklad
  2. Dodaj element na sklad
  3. Vzemni element s sklada
  4. Ali je sklad prazen?
- Vrsta:
  1. Naredi prazno vrsto
  2. Dodaj element v vrsto
  3. Vzemni element iz vrste
  4. Ali je vrsta prazna?



Tabele v Pythonu uporabimo kot sklad in kot vrsto.

1. Prazna tabela []
2. Dodamo `t.append(x)`
3. Vzamemo :
  - sklad `t.pop()`
  - vrsta `t.pop(0)`
4. Ali je prazna? `len(t) == 0`  
`bool(t)`

## Kopica ali prioriteta vrsta

Prioritetna vrsta :

$(x_0, p_0), \dots, (x_{n-1}, p_{n-1})$

podatki  $x_i$ , vsak ima prioriteto  $p_i$

Operacije :

1. Naredi prazno prioriteta vrsto
2. Dodaj  $(x, p)$  v vrsto
3. Vzemni element z najvišjo prioriteto iz vrste
4. Ali je prioriteta vrsta prazna

Ideja: Podatke hranišmo v tabeli

$[(x_0, p_0), \dots, (x_{n-1}, p_{n-1})]$

urejeni glede na  $p_i$ .

Operacije: 1. Prana  $[]$   $O(1)$

2. Dodaj:

$\log n$  - bisekcijski poišče indeks, kamor prese

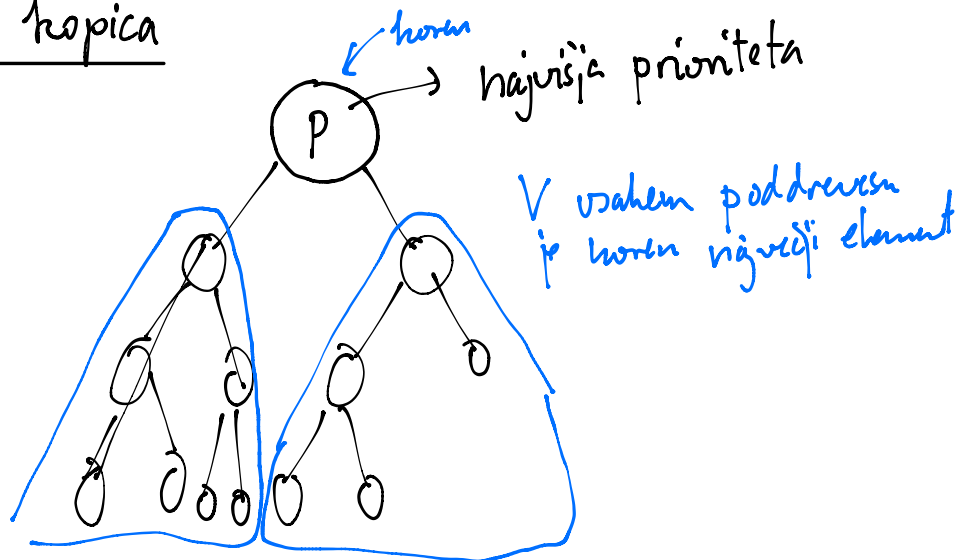
$O(n)$  - naredimo prostor za novega

$O(n)$  ← slabo...

3. Vzemi:  $O(1)$

4. Ali je prazna?  $O(1)$

Dvojška kopica



Primer:

