

# Algoritmi i strukture podataka

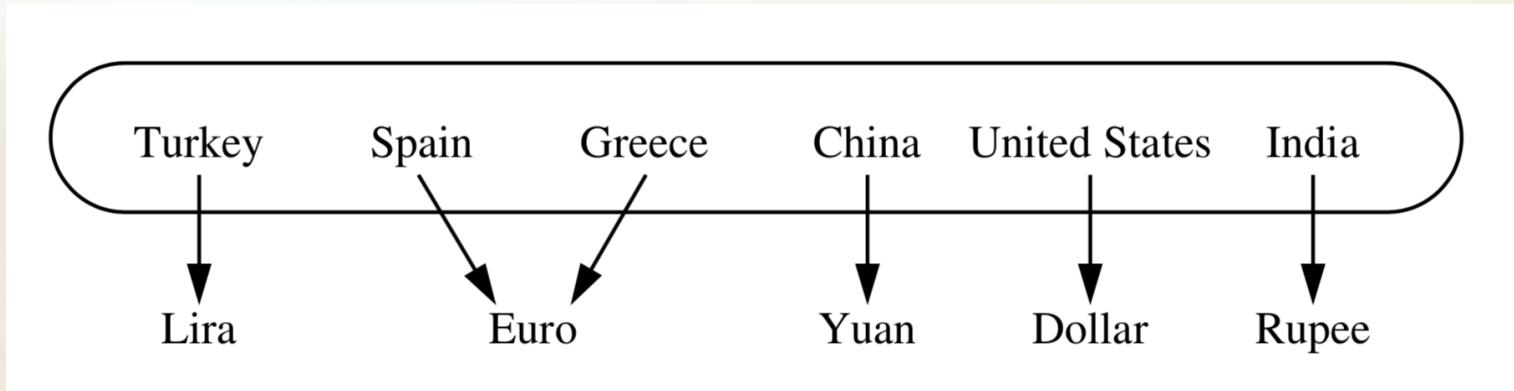
07 Hash mapa

Katedra za informatiku, Fakultet tehničkih nauka, Novi Sad

2019

# Opšte informacije

- Element se sastoji od ključa i vrednosti.
- ključevi su jedinstveni (nema ponavljanja)
- vrednosti ne moraju biti jedinstvene



# Operacije

`M[k]` vraća vrednost  $v$  vezanu za ključ  $k$  u mapi  $M$ ; ako ne postoji, izaziva `KeyError`; implementira je `__getitem__`

---

`M[k] = v` dodeljuje vrednost  $v$  ključu  $k$  u mapi  $M$ ; ako ključ već postoji, zamenjuje staru vrednost; implementira je `_setitem__`

---

`del M[k]` uklanja element sa ključem  $k$  iz mape  $M$ ; ako ne postoji, izaziva `KeyError`; implementira je `_delitem__`

---

`len(M)` vraća broj elemenata u mapi  $M$ ; implementira je `__len__`

---

`iter(M)` generiše listu **ključeva** iz mape  $M$ ; implementira je `__iter__`

# Operacije

`k in M` vraća True ako mapa  $M$  sadrži ključ  $k$ ; implementira je `__contains__`

`M.keys()` vraća skup svih ključeva iz  $M$

`M.values()` vraća skup svih vrednosti iz  $M$

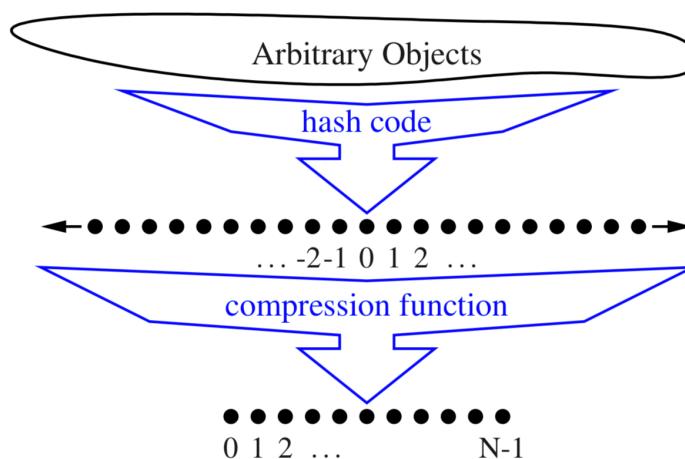
`M.items()` vraća skup svih parova  $(k,v)$  iz  $M$

# Hash funkcija

- Da bismo ubrzali pristup, mapiramo ključeve na memorijske lokacije
- Hash funkcija je svaka funkcija koja se koristi za mapiranje podataka određene dužine na podatke fiksne dužine
- Osobine:
  - Determinističnost – za iste ulaze dobijamo uvek iste izlaze
  - Uniformnost – ulazne vrednosti bi trebalo da se mapiraju na izlazni opseg na što uniformniji način. Svaka hash vrednost bi trebalo da se dobija sa istom verovatnoćom
  - Fiksan opseg
  - Kontinualnost?
  - Ne postojanje inverzne funkcije

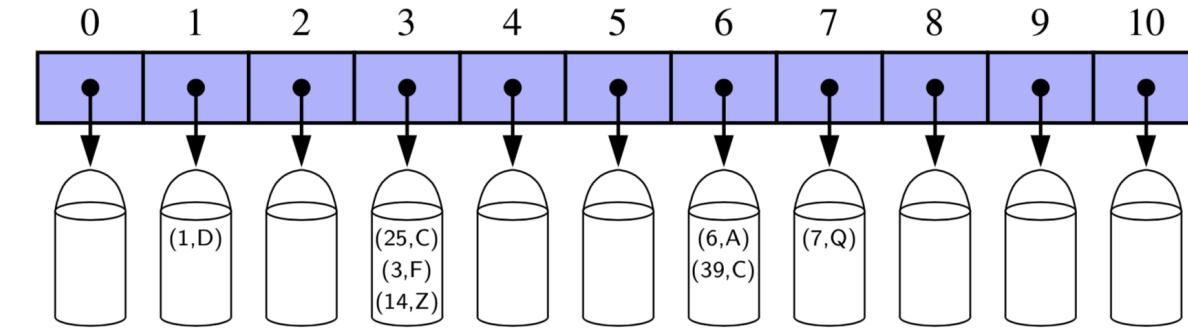
# Hash funkcija

- $\text{normalize}(\text{hash}(\text{key})) = \text{position}$ 
  - često se hash funkcija može posmatrati kao kompozicija dve funkcije:
  - **hash code**: mapira ključ na ceo broj
  - **compression function**: mapira hash kôd na broj u intervalu  $[0, N - 1]$



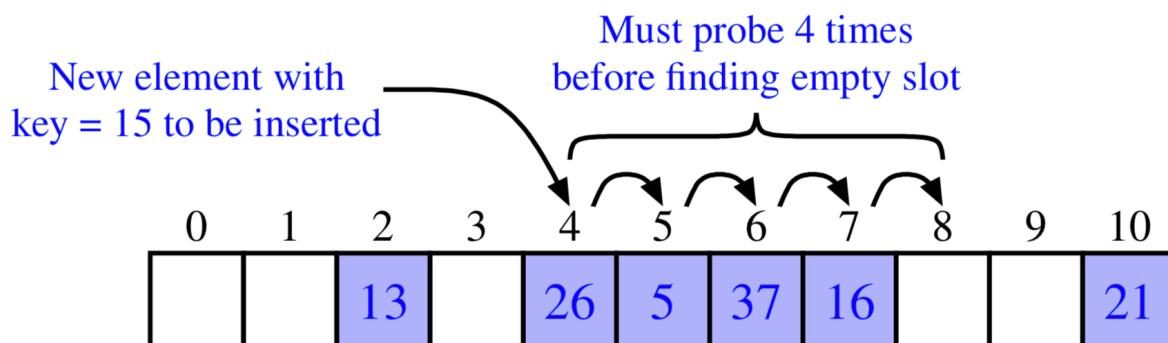
# Bucket array

- pretvorimo ključeve u cele brojeve pomoću **hash funkcije**
- dobra hash funkcija će ravnomerno distribuirati ključeve u  $[0, N - 1]$
- ali može biti duplikata
- duplike ćemo čuvati u „kantama“ – tzv. **bucket array**



# Linearno traženje

- **linear probing:** smešta element u koliziji u prvu sledeću slobodnu ćeliju (cirkularno)
- elementi u koliziji se nagomilavaju izazivajući dalje kolizije
- primer:



# Zadatak 1

- Napisati klasu **HashMap**.