

# Dictionaries

---

Igor Buzov, dipl. inf., viši predavač  
Centar umjetne inteligencije Lipik

```
country_capitals = {  
    'Germany' : 'Berlin',  
    'Canada' : 'Ottawa',  
    'England' : 'London'  
}
```

Diagram illustrating the structure of a dictionary (map) with three elements:

- element 1: 'Germany' : 'Berlin'
- element 2: 'Canada' : 'Ottawa'
- element 3: 'England' : 'London'

The keys are labeled 'key' and the values are labeled 'value'.



# Pregled lekcije

- U ovom poglavlju upoznato ćemo se s novom strukturom podataka, rječnicima
- Rječnici se sastoje od parova: ključ vrijednost

# Ponavljanje – što će biti rezultat ovog koda?

```
zemlje = ["hr", "de", "it", "fr"]
```

```
for z in zemlje:  
    print (z.upper())
```

## Ponavljanje – popravi kod

```
lista = range (2; 11; 2)
```

```
for i in lista  
    print (i)
```

## Ponavljanje – što će biti rezultat ovog koda

```
slova = ['a', 'b', 'c', 'd', 'e', 'f']
```

```
for slovo in slova:
```


```
    print("Trenutno slovo:", slovo)
```

```
    if slovo == 'c':
```

```
        print("Pronašli smo 'c', prekidamo petlju.")
```

```
        break
```

```
print("Petlja je završila.")
```



Ponavljanje – što  
će biti rezultat  
ovog koda

```
for i in range(2):  
    for j in range(3):  
        print("i =", i, ", j =", j)
```

# Objasnite algoritam binary search

Kada se koristi?

Koliko vam treba pokušaja za pronaći  
rješenje u listi od 100 članova?

Koliko vam treba pokušaja za pronaći  
rješenje u listi od 1000 članova?

Objasnite razliku između for  
petlje i while petlje

---



Otvorite novi file pod  
nazivom rjecnici.py

---

# Jednostavni dictionary - uvod

- Rječnik (eng. dictionary) – struktura podataka koja omogućuje povezivanje srodnih informacija
- Razumjevanje dictionarya nam omogućuje modeliranje raznih real world objekata
- Na primjer, kreiranjem rječnika "osoba", možemo pohraniti različite informacije o njoj (ime, starost, spol, adresa, itd)
- Rječnici su kolekcije podataka u kojima se elementi pohranjuju u obliku parova ključ-vrijednost (key i value)
- Svaki ključ (key) je spojen sa svojom vrijednošću (value) i možemo koristiti ključ za pristup pridruženoj vrijednosti
- Sadržaj rječnika se može mijenjati te nisu dozvoljeni duplikati

# Izrada jednostavnog riječnika

- Rječnik koji sadrži podatke o proizvodu:

```
proizvod = {  
    "naziv": "laptop",  
    "cijena": 1200,  
    "tezina": 1.5,  
    "dostupno": True  
}
```

```
print (proizvod)
```

- Imamo naziv rječnika, znak jednako i nakon toga niz parova omeđenih **vitičastim** zagradama
- Liste imaju uglate zagrade
- Unutar zagrada imamo kolekciju ključeva – vrijednosti
- Svaki ključ može biti broj, string pa čak i lista ili rječnik
- Svaki ključ je povezan s vrijednošću pomoću dvotočke
- Nakon svakog para slijedi znak zarez
- Uobičajeno je svaki par pisati u novom retku

# Zadatak 1

- Napravite samostalno dva nova rječnika, jedan o automobilima te jedan o voću
- Svaki vaš rječnik treba imati barem tri para key – value
- Ispišite vaše rječnike korištenje naredbe print

# Rad s rječnicima

- Za dobivanje vrijednosti koja je asocirana s određenim ključem, koristimo naziv rječnika te ključ unutar uglatih zagrada
- `print (proizvod["naziv"])`
- `print (proizvod["tezina"])`
- `print (proizvod["naziv"], proizvod["cijena"])`
- Rječnici ne podržavaju "slicing" kao što imamo s listama - `print(brojevi[0:5])`

Što će biti  
rezultat ovih  
kodova?

```
proizvod = {  
    "naziv": "laptop",  
    "cijena": 1200,  
    "tezina": 1.5,  
    "dostupno": True,  
    "cijena": 1350  
}
```

```
print (proizvod)
```

```
haljina = {  
    "boja": "crvena",  
    "veličina": 42,  
    "cijena": 130  
}
```

```
hlace = {  
    "boja": "plava",  
    "velicina": 38,  
    "cijena": 90  
}
```

```
kosarica = haljina["cijena"] +  
hlace["cijena"]
```

```
print("Ukupan iznos glasi:",  
kosarica)
```

## Dodavanje novog para key – value u rječnik

```
namjestaj = {  
    "naziv": "stol",  
    "boja": "crna"  
}  
namjestaj["duljina"] = 120  
namjestaj["sirina"] = 80  
print (namjestaj)
```

- Za dodavanje novih parova key – value u već postojeći rječnik
- Navesti naziv rječnika, key vrijednost u uglatim zagradama i value nakon znaka jednakosti

# Promjena vrijednosti postojećeg para key - value

```
namjestaj = {  
    "naziv": "stol",  
    "boja": "crna"  
}  
print (namjestaj)  
namjestaj["boja"] = "bijela"  
print (namjestaj)
```

- Promjena vrijednosti na postojećem paru:
  - Navesti naziv rječnika s ključem čiju vrijednost želimo mjenjati
  - Znak jednakosti
  - Nova vrijednost





## Zadatak 2

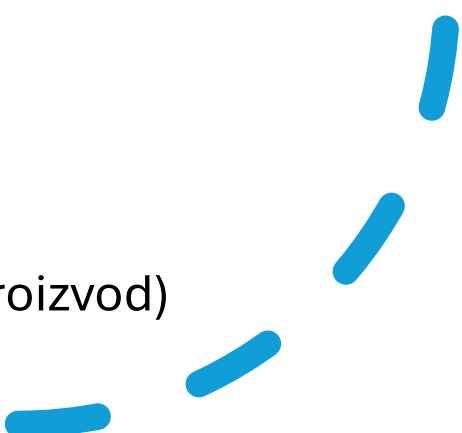
- Napravite novi rječnik po izboru s nekoliko parova key- value
- Dodajte naknadno u rječnik još jedan par key – value
- Ažurirajte jednu od postojećih vrijednosti

# Brisanje stavke iz rječnika– del naredba

- Del naredbu koristimo kada želimo obrisati stavku rječnika na osnovi njenog ključa
- Ako ne postoji ključ, pojavit će se poruka o grešci

```
proizvod = {  
    "naziv": "Laptop",  
    "cijena": 1200.00,  
    "tezina": 1.5,  
    "dostupno": True  
}
```

```
print("Originalni rječnik:", proizvod)  
# Brisanje ključa 'tezina'  
del proizvod["tezina"]  
print("Nakon brisanja 'tezina' pomoću del:", proizvod)
```

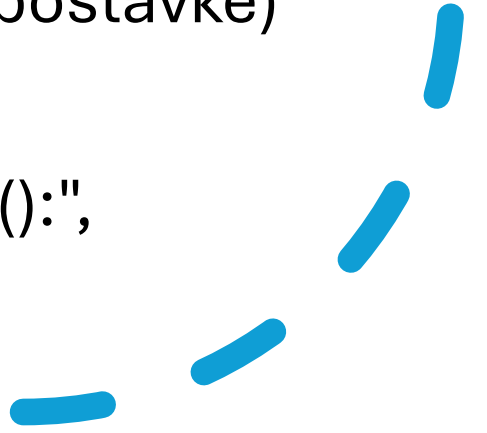


# Brisanje pomoću clear() metode (briše sve)

- Metoda .clear() uklanja sve parove ključ-vrijednost iz rječnika, čineći ga praznim.

```
postavke = {  
    "jezik": "hrvatski",  
    "tema": "tamna",  
    "notifikacije": True  
}
```

```
print("Originalni rječnik postavki:", postavke)  
postavke.clear()  
print("Rječnik postavki nakon clear():",  
postavke)
```



# Brisanje više stavki iz rječnika

- Nije moguće brisanje više stavki iz rječnika korištenjem slicinga, kao u listama
- Za brisanje više elemenata, koriste se for petlje koje prolaze kroz nazive ključeva koji će se obrisati

```
d = {'a': 1, 'b': 2, 'c': 3, 'd': 4}
```

```
# Keys to remove
```

```
key_rmv = ['b', 'd']
```

```
# Remove keys using del in a loop
```

```
for key in key_rmv:
```

```
    if key in d: # Check if the key exists
```

```
        del d[key]
```

```
print(d)
```



## Zadatak 3

- Iz vašeg rječnika iz prošlog zadatka obrišite jednu stavku



# Korištenje naredbe get() za pristup vrijednostima

- Ako želimo dobiti vrijednost za ključ koji nije definiran u rječniku, dobit ćemo poruku o grešci

```
proizvod = {  
    "naziv": "Laptop",  
    "cijena": 1200.00,  
    "tezina": 1.5,  
    "dostupno": True  
}  
print(proizvod["marka"])
```

# Korištenje naredbe get() za pristup vrijednostima

```
proizvod = {  
    "naziv": "Laptop",  
    "cijena": 1200.00,  
    "tezina": 1.5,  
    "dostupno": True  
}  
odgovor = proizvod.get("marka", "Nemamo odgovor")  
print (odgovor)
```

- Metoda get() ima dva parametra
  - Prvi parametar je **keyname**, ime ključa za koji želimo odbiti vrijednost (obavezno)
  - Drugi parametar je **value**, odgovor koji ćemo dobiti ako taj ključ ne postoji (nije obavezno=

# Primjer 1

```
favorite_languages = {  
    'jen': 'python',  
    'sarah': 'c',  
    'edward': 'rust',  
    'phil': 'python',  
}  
  
ime = input("Unesite ime osobe čiji najdraži programski jezik  
želite saznati? ").lower()  
  
if ime in favorite_languages:  
    jezik = favorite_languages[ime]  
    print ("najdraži jezik te osobe je: ", jezik)  
else:  
    print ("Osoba nije na popisu")
```



# Zadatak 4

- Napravite rječnik s podacima o studentu (ime, prosjek ocjena, broj položenih ispita)
- Napravite program koji provjerava zadovoljava li student uvjete za stipendiju
- Uvjeti (oba moraju biti ispunjena)
  - Broj položenih ispita: veći ili jednak 10
  - Prosjek ocjena: veći ili jednak 4.5
- Ispisuju se poruke: stipendija odobrena za studenta (njegovo ime) ili stipendija odbijena

## Zadatak 5 – napravite ova dva rječnika

```
kupacA = {  
    "ime": "Ana",  
    "ima_karticu_vjernosti": True,  
    "ukupna_potrosnja": 50.0,  
    "broj_kupovina": 2  
}
```

```
kupacB = {  
    "ime": "Marko",  
    "ima_karticu_vjernosti": False,  
    "ukupna_potrosnja": 150.0,  
    "broj_kupovina": 7  
}
```

# Zadatak 5 – izrada programa

- Napraviti ćete program prvo za kupcaA, a poslije ga modificirati za kupcaB
- Želite vidjeti ostvaruje li kupac pravo na SUPERPOPUST
- Uvjeti su:
  - Ima karticu vjernosti (ima\_karticu\_vjernosti je True) Ili je imao ukupna\_potrošnja veću od 100 EUR I broj\_kupovina je veći od 5.
- Ispisuje se poruka na kraju, Ostvaruje se superpopust ili se ne ostvaruje

# Zadatak 6 – virtualni aparat za pića

Grupni rad

Po uputama koje ćete dobiti, napraviti aplikaciju za virtualni aparat za pića