Datoteke

Zašto datoteke?

Temeljni pojmovi

Vrste datoteka po sadržaju

Podatkovne datoteke

Struktura podatkovne datoteke

Datoteke u Python-u

Zašto datoteke ? – Primjer 1.

- Primjer
  1. Izvršite popis učenika u razredu.
  2. Unesite popis u računalo.
  3. Izvršite SORT po abecednom redu.
  4. Tiskajmo izvještaj.
- PROBLEM !!!
  - Došao/ otišao učenik !
- Što učiniti ?
  - Očito - Ponoviti radnje po stavkama 1.; 2.; 3. i 4.
- DA LI JE OVO UČINKOVITO ?? (Sigurno nije !)

Primjer 2: Osobni karton djelatnika

- Podaci koji se vode - evidentiraju u osobnom kartonu:
  - Prezime
  - Ime
  - Adresa stanovanja
  - Ulica i broj
  - mjesto
  - Jedinstveni matični broj građanina
  - Telefonski broj

Temeljni pojmovi

- Datoteka (eng. File):
  - Organizirani skup podataka koji se obrađuju kao cjelina i pospremaju u memoriju računalnog sustava.

- Podatak (eng. Date):
  - Podatak je činjenica predočena u formaliziranom obliku, npr. broj, riječ ili slika.
  - Formalizirani znakovni prikaz činjenica, pojmova i instrukcija pogodan za priopćavanje, interpretaciju ili obradu

Datoteka - sadržaj, vrste

- Datoteka - Tekstualni sadržaj (DOC, TXT, …)
- Datoteka - Programska (BAS, PAS, FOR, …)
- Datoteka - Slika (WMF, GIF, BMP, ICO, …)
- Datoteka - Animacija (AVI, JPG, MOV, …)
- Datoteka - Zvuk (WAW, …)
- Datoteka - Arhivska (ZIP, ARJ, BAK, …)

- Datoteka čiji su sadržaj podaci je predmet rasprava i analiza ovog dijela izlaganja je - **podatkovna datoteka**

Podatkovna datoteka

- Definicija:
  - Skup podataka koji mogu biti predmet obrade jednog ili više programa u nekom programskom jeziku.
- Struktura:
  - Datoteka se dijeli na zapise (eng. record), a zapisi na polja (eng. field)
  - Logička organizacija:
    - Skup logičko-semantičkih zapisa koji se odnose na određeni niz pojmova iste vrste.
    - Svaki logički zapis u datoteci ima isti opis i isti redoslijed polja podataka u svojoj strukturi.
  - Fizička organizacija:
    - Skup fizičkih zapisa na fizičkom nosiocu podataka (masovnom memorijskom mediju, radnoj memoriji)

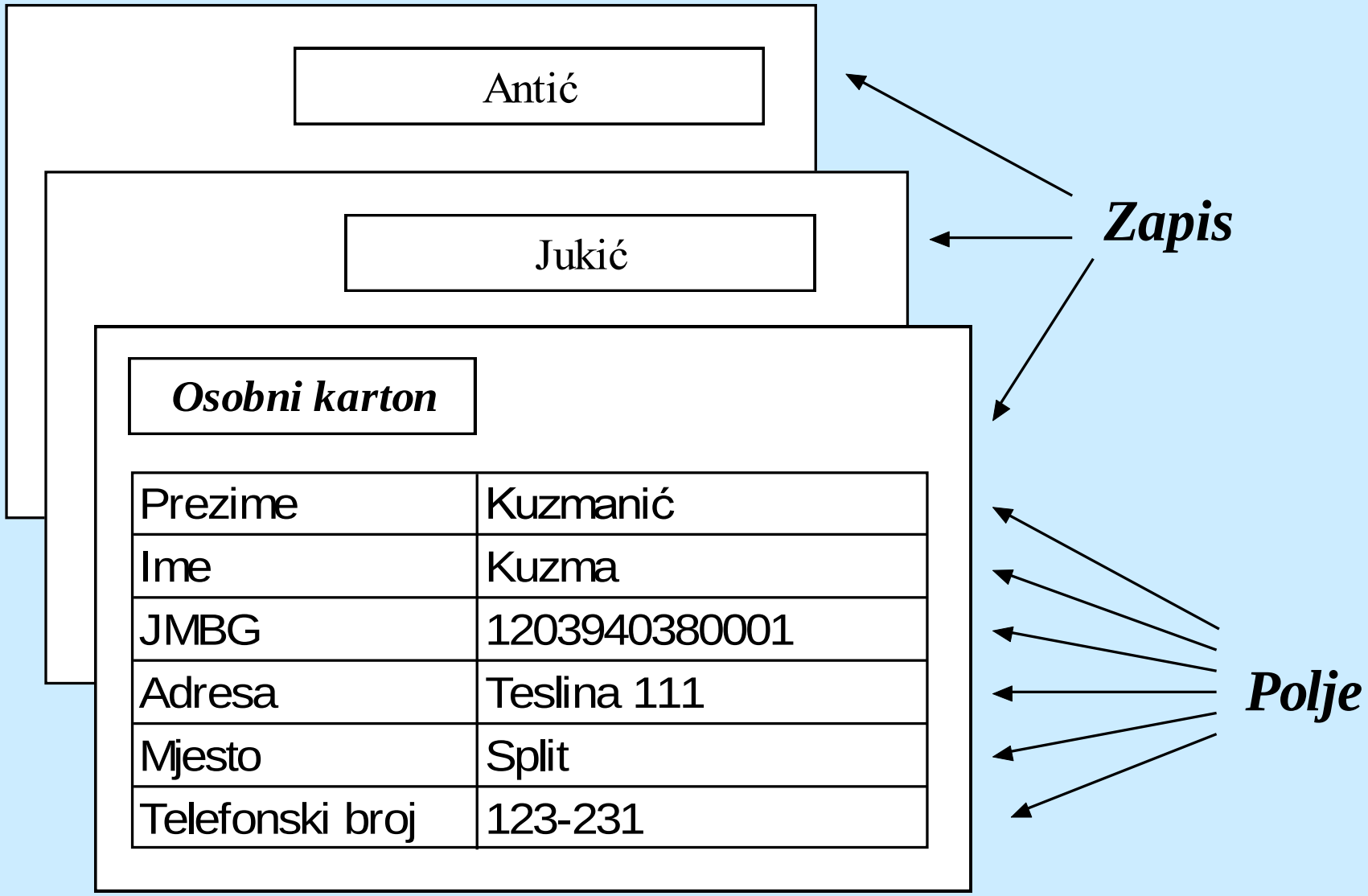
Datoteka – Tablica s podacima

- Kada bismo datoteke usporedili s tablicama tada su zapisi retci, a polja stupci.
- Npr. Ucenik

Ime	Ocjena
Ivan Ivanić	4.4
Anica Martinović	4.3

  - Ucenik - naziv strukture podataka
  - Ime, Ocjena - polja zapisa (stupci, atributi)

Podatkovna datoteka - Struktura



Tipični procesi - procedure za obradu podataka

- Unos podataka
- Ažuriranje podataka
  - Brisanje
  - Promjene
- Prikazi podataka
  - Izvještaji
  - Traženja

Postupan je s datotekama

- U različitim programskim jezicima se s datotekama radi na različit način, ali su osnovni koncepti jednaki

- Otvaranje datoteke
  - Povezivanje datoteke na disku s varijablom u programu
  - Jednom kad je datoteka otvorena, pomoću varijable unutar programa možemo čitati podatke iz datoteke, te upisivati podatke u datoteku

- Zatvaranje datoteke
  - Omogućuje završetak upisa podataka u datoteku
  - Promjene u datoteci na disku će biti vidljive tek kad se datoteka zatvori

Datoteke u Python-u

- Varijabla može biti i tipa *datoteka,* te kao takva može imati razne operacije koje se odnose na datoteke (operacije čitanja, pisanja, ...)
- Prvi korak je povezivanje varijable u programu s datotekom na disku, moramo pozvati posebnu naredbu
- To ćemo napraviti naredbom: **open**

```
<varijabla> = open(<naziv>, <nacin>)
```

Naziv (ime) datoteke na disku

String koji označava način otvaranja datoteke:
"r" → čitanje (read)
"w" → pisanje (write)
"a" → dodavanje (append)

Zatvaranje datoteke

- Datoteku moramo obavezno zatvoriti nakon čitanja
- Ako je ne zatvorimo, može se dogoditi da joj drugi programi ne mogu pristupiti

- To se radi operacijom **close**:

```
<varijabla>.close()
```

Upis u datoteku

- Otvaranje datoteke za pisanje priprema datoteku za primanje podataka
- Ako datoteka s navedenim imenom ne postoji, stvorit će se nova datoteka
- Ako datoteka s navedenim imenom postoji, onda će se prebrisati i stvoriti nova prazna datoteka!

- Primjer:
- Instrukcija pomoću koje se otvara datoteka za pisanje

```
dat = open("izlaz.txt", "w")
```

Instrukcija write()

<varijabla>.write(<string>)

- **write** radi slično kao i **print**, ali je malo manje fleksibilna
- Uzima samo jedan parametar koji mora biti **string**, te ga upisuje u datoteku
- To može biti i rezultat neke operacije (npr. spajanja +), ali mora biti samo jedan string
- Ako želite početi novu liniju u datoteci, onda se mora upisati i oznaka za novi red (*new line character*):

\n

- Primjer: Datoteke – Prog_1.py

QBasic Datoteke

Za snimanje podataka u datoteku:

1. Pozovi **OPEN** komandu (ime datoteke, modalitet datoteke OUTPUT, logički broj datoteke)
2. Upotrijebi **PRINT** komandu zajedno s logičkim brojem datoteke kako bi zapisao podatke u datoteku
3. Zatvori datoteku sa **CLOSE** komandom

```
OPEN "test.dat" FOR OUTPUT AS #1
PRINT #1, "Pozdra  Svima"
CLOSE #1
```

Kod imena datoteke se može navesti i cijela putanja, npr. "c:\temp\datoteka.dat"
Ako putanja nije navedena, onda se datoteka snimila u direktorij u kojem se nalazi QBasic

Datoteke

Za čitanje podataka iz datoteke:

1. Pozovi **OPEN** komandu (ime datoteke, modalitet datoteke INPUT, logički broj datoteke)
2. Upotrijebi **INPUT** komandu zajedno s logičkim brojem datoteke kako bi zapisao podatke iz datoteke u varijablu
3. Zatvori datoteku sa **CLOSE** komandom

```
OPEN "test.dat" FOR INPUT AS #1
INPUT #1, tekst$
CLOSE #1
PRINT tekst$
```

```
Pozdrav svima
```

Unos i čitanje

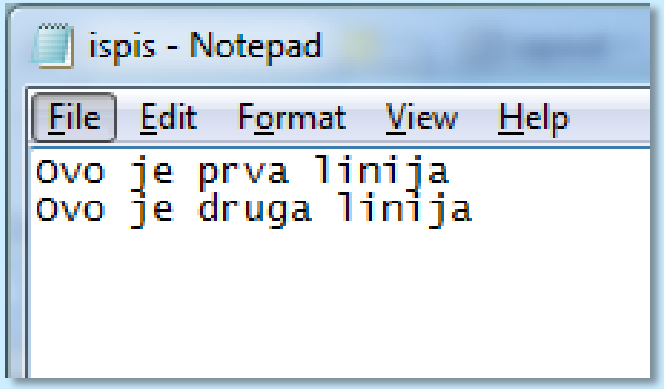
```
    REM Unos podataka
    CLS
    INPUT "Unesi naziv datoteke : "; N$
    OPEN N$ FOR OUTPUT AS #1
11  INPUT "Unesi zapis"; a$
    PRINT #1, a$
            INPUT "Nastavak D/N ??"; O$
            IF UCASE$(O$) = "D" THEN
            GOTO 11
    END IF
    CLOSE #1
    REM Citanje podataka
    CLS
    INPUT "Unesi naziv datoteke s podacima : "; N$
    OPEN N$ FOR INPUT AS #1
    DO WHILE NOT (EOF(1))
            INPUT #1, a$
            PRINT a$
    LOOP
    CLOSE #1
```

Instrukcija write()

```
dat = open("ispis.txt", "w")

dat.write("Ovo je prva linija\n")
dat.write("Ovo je druga linija\n")

dat.close()
```



Čitanje iz datoteke

- U Python-u postoje tri operacije za čitanje:

```
<varijabla>.read()
<varijabla>.readline()
<varijabla>.readlines()
```

Instrukcija read()

- Operacija **read** vraća cijeli sadržaj datoteke kao jedan string
- Ako postoji više linija teksta u datoteci, na kraju svake linije je “oznaka za novi red”
- Tekst:
  - Danas ucimo datoteke.
    Ovo je druga linija,
    a ovo treca.
- se može interpretirati ovako:
  - Danas ucimo datoteke.\nOvo je druga linija, \na ovo treca. \n

\n

New line character

\n ne vidimo na ekranu u programima za uređivanje teksta

Primjer: Datoteke – Prog_1.py

```
imedat = input("Upisi ime datoteke:")
dat = open(imedat, "r")

podaci = dat.read()

print (podaci)

dat.close()
```

Instrukcija readline()

- Operacijom **readline** čitamo jednu liniju iz datoteke
- Čitaju se svi znakovi do oznake za kraj linije: **\n**
- Svakim sljedećim pozivom, čita se sljedeća linija
- Ovdje moramo zapamtiti da se sa svakim čitanjem pročita i oznaka za kraj linije

```
dat = open("primjer.txt", "r")
for i in range(3):
    linija = dat.readline()
    print(linija)
dat.close()
```

Čitanje i tiskanje podataka iz datoteke pomoću instrukcije readline()

- Kod ispisa na ekran, **`print`** uvijek nakon svakog ispisa prelazi u novi red, pa ako varijabla *linija* već sadrži jedan \n, onda će ispis izgledati ovako:

```
Ovo je tekstualna datoteka

koja se sastoji od

3 linije teksta
```

- Kako bi to izbjegli, moramo ukloniti zadnji znak iz varijable *linija,* npr. ovako:

```
print(linija[:-1])
```

Instrukcija readlines()

```
dat = open("primjer.txt", "r")
sve_linije = dat.readlines()

print("Broj linija:", len(sve_linije))

for linija in sve_linije:
    print(linija[:-1])

dat.close()
```

```
Broj linija: 3
Ovo je tekstualna datoteka
koja se sastoji od
3 linije teksta
```

Čitanje pomoću for petlje...

- Može malo i jednostavnije

```
datoteka = open("primjer.txt", "r")

for linija in datoteka:
    print(linija[:-1])

dat.close()
```

Unos brojeva u datoteku

- Ako se pomoću **write()** mogu unositi samo stringovi, onda brojeve moramo pretvoriti u string

```
dat.write(str(broj))
```

- Možemo i zalijepiti “\n” jer inače brojevi neće biti odvojeni u datoteci

```
dat.write(str(broj)+ "\n")
```

- Kod čitanja, ako želimo s tim brojevima računati, onda ih moramo pretvoriti u odgovarajući tip

```
for linija in dat:
    broj = int(linija)
    print(broj,"na kvadrat:",broj**2)
```

Pretvorbom će se \n odbaciti

Ponovimo...

- Sve što unosimo s **`input()`** je tipa string
  - Prije upisa u datoteku, brojeve moramo pretvoriti u string funkcijom **`str()`**
  - Ako želimo unijeti brojeve s kojima moramo nešto računati, onda ono što smo unijeli s **`input`** pretvaramo funkcijama: **`int()`, `float()`** i **`eval()`**

- **`Primjer: Datoteke – Prog_2.py`**
  - **`Generiranje slučajnih brojeva u zadanim granicama`**
  - **`Otvori datoteku za pisanje - „w”`**
  - **`Otvori datoteku za čitanje – „r”`**
  - **`Ispiši brojeve (stupac, redak)`**
  - **`Nađi min i max broj`**

Pseudo kod za unos i čitanje podataka

```
Unos (datoteka)
Otvori datoteku za unos
Ponavljaj
       Unos (zapis)
       Unos u datoteku (zapis)
       Unos (odgovor)
Dok je odgovor<>"N"
Zatvori datoteku

Unos (datoteka)
Otvori datoteku za čitanje
Dok nije kraj datoteke
       Unos iz datoteke (zapis)
       Ispis (zapis)
Ponavljaj
Zatvori datoteku
```

Primjer: Datoteke – Prog_3.py

Python program – Unos - Čitanje

```
ime = input("Unesi ime datoteke: ")
dat = open(ime, "w")

#unos u datoteku
while True:
    tekst = input("Unesi zapis: ")
    dat.write(tekst)
    #ako zelimo prijelaz u novu liniju:
    dat.write("\n")

    odg = input("Zelite li nastaviti (D/N): ")
    if odg == "N" or odg == "n":
        break

dat.close()
```

```
#citanje
print("Ispis sadrzaja datoteke:")
dat = open(ime, "r")

for linija in dat:
    print(linija[:-1])

dat.close()
```