# OSNOVE PROGRAMIRANJA U PAJTONU

## PREDAVANJE 13: TEKSTUALNE DATOTEKE

Miloš Kovačević

Đorđe Nedeljković

Marija Petronijević

Dušan Isailović

# SADRŽAJ PREDAVANJA

- Osnovni pojmovi
- Sistem datoteka
- Tekstualne datoteke – čitanje i pisanje

# POJAM DATOTEKE

Objekti, kojima su apatrahovani podaci u programu,
smešteni su tokom izvršavanja programa u operativnu memoriju (OM).

Sadržaj OM gubi se po prestanku napajanja pa podatke treba smestiti u pogodnom formatu u trajnu memoriju (npr. HDD, SDD)

Datoteka – imenovani skup podataka smešten u trajnoj memoriji.

Podaci u datoteci odnose se na informacije koje potiču iz istog konteksta.

Ime datoteke je tekstualna sekvenca oblika `s1.s2 (merge_sort.py)`
`s2` predstavlja ekstenziju koja asocira na prirodu podataka u datoteci.

# BINARNE I TEKSTUALNE DATOTEKE

Prema tome kako program tumači bite koji čine datoteku,
dele se na binarne i tekstualne.

Binarna datoteka ima predefinisanu strukturu koju određuje programer:
program koji je obrađuje mora da poznaje značenje pojedinih grupa bita.

Primer:

datoteka koja počinje sa *k* bajtova koji označavaju broj zapisa o studentima,
pri čemu je, za informacije o svakom studentu (jedan zapis), potrebno *n* bajtova.
Zapisi o studentima slede posle informacije o broju studenata.

Tekstualna datoteka: niska bita predstavlja redove teksta
razdvojenih specijalnim simbolom za novi red.
Redovi teksta – niske bita koje označavaju pojedinačne karaktere
(npr. po Unicode standardu).

# SISTEM DATOTEKA

Datoteke trajne memorije organizovane su u sistem datoteka – hijerarhijska organizacija direktorijuma.

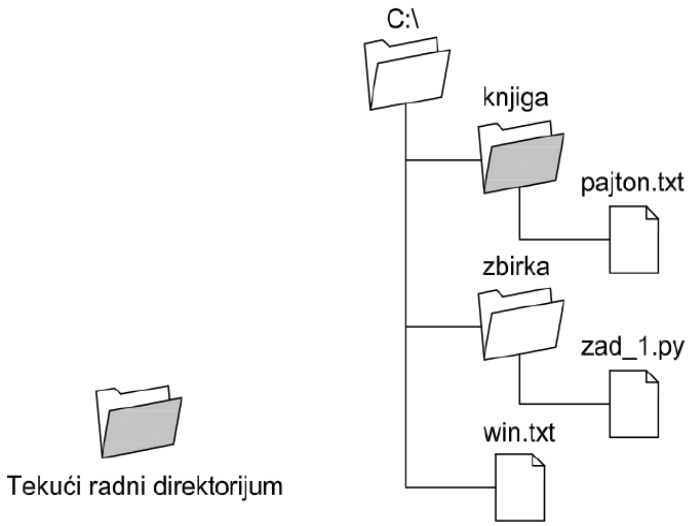
Direktorijumi, pored datoteka, mogu da sadrže i druge direktorijume (poddirektorijumi).

Direktorijum koji u spomenutoj hijerarhiji nije ujedno i poddirektorijum – koreni direktorijum.

Sistem datoteka na Windows-u može imati više korenih direktorijuma: C:\, D:\, …

# APSOLUTNE I RELATIVNE PUTANJE

Datoteka je jedinstveno određena apsolutnom putanjom u sistemu datoteka:
niz imena svih direktorijuma, počevši od korenog
pa do onog koji je neposredno sadrži + ime datoteke.
Imena u putanji razdvojena separatorom (na Windows-u \ )




| Relativne putanje | Apsolutne putanje |
|---|---|
| .. | C:\ |
| . | C:\knjiga |
| pajton.txt | C:\knjiga\pajton.txt |
| ..\zbirka | C:\zbirka |
| ..\zbirka\zad_1.py | C:\zbirka\zad_1.py |
| ..\win.txt | C:\win.txt |

Relativna putanja: imena direktorijuma počevši od tekućeg radnog direktorijuma + ime datoteke.

# SISTEM DATOTEKA – MODUL `os`

| funkcija | opis |
|---:|---|
| `os.getcwd()` | vraća putanju tekućeg direktorijuma |
| `os.chdir(p)` | postavlja tekući direktorijum zadat putanjom p |
| `os.listdir(p)` | vraća listu direktorijuma i datoteka iz direktorijuma sa putanjom p |
| `os.mkdir(p)` | kreira direktorijum zadat putanjom, ako ne postoji |
| `os.rmdir(p)` | uklanja direktorijum zadat putanjom, ako je prazan |
| `os.mkdirs(p)` | kreira sve direktorijume iz putanje, ako ne postoje |
| `os.rmdirs(p)` | uklanja sve direktorijume iz putanje, ako su prazni |
| `os.remove(p)` | uklanja datoteku zadatu putanjom |

# MODUL os

Specijalna sekvenca za separator imena: \\

```
>>> import os
>>> os.getcwd()  # tekući radni direktorijum
'C:\\Users\\Andjelko\\AppData\\Local\\Programs\\Python\\Python35'
>>> os.chdir('c:\\papers')  # promeni tekući radni direktorijum
>>> os.getcwd()
'c:\\papers'
>>> os.listdir()  # izlistaj sadržaj radnog dir.
['2016', '2017']
>>> os.listdir('c:\\papers\\2017')  # izlistaj zadati dir.
['jcce']
>>> os.mkdir('c:\\papers\\a')  # napravi zadati dir.
>>> os.listdir('c:\\papers)
['2016', '2017', 'a']
>>> os.rmdir('c:\\papers\\a')  # ukloni dir. (samo ako je prazan!)
>>> os.listdir('c:\\papers')
['2016', '2017']
>>> # napravi dir. i sve iz putanje, ako treba
>>> os.makedirs('c:\\papers\\a\\b\\x')
>>> # ukloni sve prazne dir. iz putanje počev od x!
>>> os.removedirs('c:\\papers\\a\\b\\x')
>>> os.listdir('c:\\papers')
['2016', '2017']
>>> # pošto je napravljena nova datoteka iz Windows-a
>>> os.listdir('c:\\papers')
['2016', '2017', 'novi.txt']
>>> os.remove('c:\\papers\\novi.txt') # ukloni datoteku
>>> os.listdir('c:\\papers')
['2016', '2017']
```

# MODUL `os.path`

| funkcija | opis |
|---:|---|
| `os.path.abspath(p)` | vraća apsolutnu putanju za `p` |
| `os.path.isabs(p)` | ispituje da li je `p` apsolutna |
| `os.path.relpath(p)` | vraća relativnu putanju za `p` |
| `os.path.dirname(p)` | vraća putanju direktorijuma za `p` |
| `os.path.basename(p)` | vraća ime datoteke za `p` |
| `os.path.join(f1,...,fn)` | pravi putanju od tekstualnih fragmenata `f1`,...,`fn` |
| `os.path.getsize(p)` | vraća veličinu datoteke (direktorijuma) u bajtima |
| `os.path.getctime(p)` | vraća sistemsko vreme nastanka `p` |
| `os.path.getmtime(p)` | vraća sistemsko vreme poslednjeg ažuriranja `p` |
| `os.path.getatime(p)` | vraća sistemsko vreme poslednjeg pristupanja `p` |
| `os.path.exists(p)` | ispituje da li `p` postoji u sistemu datoteka |
| `os.path.isfile(p)` | ispituje da li je `p` datoteka |
| `os.path.isdir(p)` | ispituje da li je `p` direktorijum |

# MODUL os.path

```
>>> import os.path as osp
>>> os.chdir('C:\\') # promena za tek. rad. dir.
>>> osp.isabs('c:\\Users')
True
>>> osp.abspath('a\\b\\c.txt') # u odnosu na tek. rad. dir.
'C:\\a\\b\\c.txt'
>>> osp.dirname('C:\\a\\b\\c.txt'), osp.basename('C:\\a\\b\\c.txt')
('C:\\a\\b', 'c.txt')
>>> osp.join('a','b','c.txt')  # spaja fragmente putanje.
'a\\b\\c.txt'
>>> osp.exists('C:\\nepostoji.txt')
False
>>> osp.isfile('devlist.txt'), osp.isdir('devlist.txt')
(True, False)
>>> osp.getctime('devlist.txt'), osp.getmtime('devlist.txt')
(1455880136.8817973, 1455879611.5920193)
>>> osp.getsize('devlist.txt')
12019
```

Broj sekundi
u odnosu na epohu

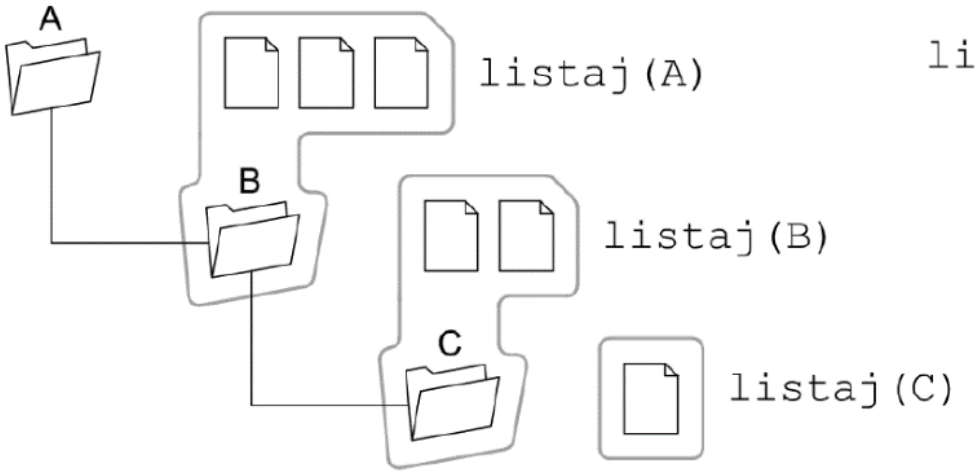
**Problem 9.1 — Stablo direktorijuma.** Kreirati funkciju koja prikazuje informacije o datotekama koje se nalaze u i ispod navedenog direktorijuma, u hijerarhiji sistema datoteka. Prikazati informacije o veličini i vremenu kreiranja svake datoteke. ■

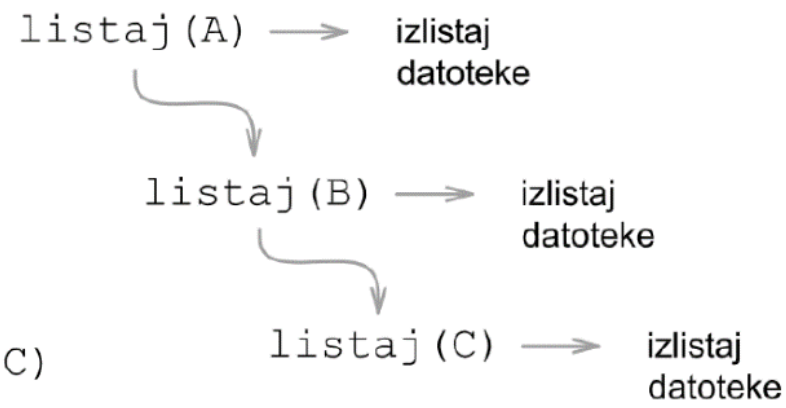
```
unesite putanju direktorijuma: c:\papers\2017
+ c:\papers\2017\jcce
+-- jcce.7z                15752K Wed Jun 21 19:45:26 2017
+-- oldPaper.docx           3233K Wed Jun 21 19:45:26 2017
+-- paper.rar              21028K Wed Jun 21 19:45:26 2017
+-- response.doc              75K Wed Jun 21 19:45:26 2017
+-- revised.docx              96K Wed Jun 21 19:45:26 2017
+-- revised2.docx             95K Wed Jun 21 19:45:26 2017
+-- c:\papers\2017\jcce\slikePdf
+---- Fig. 1.pdf               30K Wed Jun 21 19:45:26 2017
+---- Fig. 10.pdf             135K Wed Jun 21 19:45:26 2017
+---- Fig. 11.pdf             326K Wed Jun 21 19:45:26 2017
+---- Fig. 2.pdf               48K Wed Jun 21 19:45:26 2017
+---- Fig. 3.pdf               49K Wed Jun 21 19:45:26 2017
+---- Fig. 4.pdf               17K Wed Jun 21 19:45:26 2017
+---- Fig. 5.pdf               53K Wed Jun 21 19:45:26 2017
+---- Fig. 6.pdf             2235K Wed Jun 21 19:45:26 2017
+---- Fig. 7.pdf               42K Wed Jun 21 19:45:26 2017
+---- Fig. 8.pdf            15114K Wed Jun 21 19:45:26 2017
+---- Fig. 9.pdf              185K Wed Jun 21 19:45:27 2017
+ test.txt                   1K Fri Jun 23 13:28:44 2017
 Ukupno 58514K
```

# STABLO DIREKTORIJUMA

Posetiti sve datoteke iz direktorijuma korišćenjem njihovih rel. putanja

Ako direktorijum sadrži i poddirektorijume – ponoviti postupak (rekurzija)




Za crtanje stabla direktorijuma potrebna informacija o dubini,
u odnosu na zadati početni direktorijum (dubina 0)!

Funkcija u funkciji da sakrije detalj implementacije (`dubina`)

```
import os.path as p
import math as m
import time as t

def listaj(ul_dir):

    def listaj_rek(ul_dir, dubina):
        total = 0
        for d in os.listdir(ul_dir):
            d = p.join(ul_dir, d)
            if p.isdir(d):
                print('+{} {}'.format(dubina * '--', d))
                total += listaj_rek(d, dubina + 1)
            elif p.isfile(d):
                veličina = m.ceil(p.getsize(d)/1024)
                total += veličina
                print('+{} {:20}\t{:6}K\t{}'.format(
                    dubina * '--',
                    p.basename(d),
                    veličina,
                    t.asctime(t.localtime(p.getctime(d)))))

        return total
    return listaj_rek(ul_dir, 0)
```

# ČITANJE CELOG SADRŽAJA TEKSTUALNE DATOTEKE

Datoteka se pre čitanja (upisivanja) mora otvoriti.

```
>>> d = open('c:\\papers\\test.txt')  # otvoreno za čitanje
>>> type(d)
<class '_io.TextIOWrapper'>
```

```
>>> d.read()  # čita ceo sadržaj
'Ja znam sva tvoja lica, svako šta hoce, šta nosi,\ngledao sam
 sve tvoje oci, razumem šta kažu, šta kriju.\nja mislim tvoju
 misao za celom ti u kosi,\nja znam tvoja usta šta ljube, šta piju.'
>>>
>>> d.read()  # pročitana cela datoteka, vraća prazan tekst
''
>>> d.close() # zatvara datoteku
>>> d.read()  # posle zatvaranja, nema čitanja
Traceback (most recent call last):
  File "<pyshell#102>", line 1, in <module>
    d.read()
ValueError: I/O operation on closed file.
```

Separator za novi red.
(nema ga iza poslednjeg reda)

Posle čitanja (pisanja) datoteka se obavezno zatvara,
kako bi se olobodili resursi operativnog sistema.

# ČITANJE IZ TEKSTUALNE DATOTEKE RED PO RED

Ako je datoteka velika ili ne može stati u radnu memoriju, čita se red po red.

```
>>> d = open('c:\\papers\\testSrpski.txt', encoding='utf-8')
>>> for l in d:
        print(l)

Ja znam sva tvoja lica, svako šta hoće, šta nosi,

gledao sam sve tvoje oči, razumem šta kažu, šta kriju.

Ja mislim tvoju misao za čelom ti u kosi,

ja znam tvoja usta šta ljube, šta piju.
>>> d.close()
```

Opcioni parametar za kodiranje
(preporučuje se snimanje u Unicode UTF-8 formatu)

# UPISIVANJE U TEKSTUALNU DATOTEKU

Pri upisu, sadržaj postojeće datoteke može se prepisati, ili se može dodavati na kraj.

```
>>> # upisivanje, kreira novu datoteku
>>> d = open('c:\\papers\\nova.txt', 'w')
>>> d.write('Prvi red\nDrugi red')
18
>>> d.close()
>>> # dodavanje
>>> d = open('c:\\papers\\nova.txt', 'a')
>>> d.write('\nTreci red')
10
>>> d.close()
>>> # čitanje
>>> d = open('c:\\papers\\nova.txt') # opciono 'r' se podrazumeva!
>>> print(d.read())
Prvi red
Drugi red
Treci red
>>> d.close()
```

Pri upisu, korisnik sam vodi računa o separatoru novog reda.

# UPISIVANJE U TEKSTUALNU DATOTEKU

```
>>> # upisivanje, briše stari sadržaj
>>> d = open('c:\\papers\\nova.txt', 'w')
>>> d.write('Cetvrti red\n')
12
>>> d.close()
>>> # čitanje
>>> d = open('c:\\papers\\nova.txt')
>>> d.read()
'Cetvrti red\n'
>>> d.close()
```

# NAREDBA `with`

Naredba `with` omogućava da se datoteka otvori pod želejnim imenom, obave ulazno izlazne operacije i potom sve datoteke automatski zatvore kada se napusti blok ove naredbe.

```
with open('c:\\papers\\proba.txt', 'a') as f:
    f.write('\ndodaj ovaj red')

with open('c:\\papers\\proba.txt') as f:
    print(f.read())
```

Datoteka se zatvara u svim slučajevima (čak i kada se desi greška, ili naiđe na `return` u bloku naredbe `with`)

Naredba `with` omogućava istovremeno otvaranje više datoteka.

**Problem 9.2 — Kopiranje.** Kreirati funkciju koja kopira sadržaj tekstualne datoteke, uz adekvatnu obradu mogućih izuzetaka. ■

```
import os
def kopiraj(ulaz, izlaz, kodiranje='utf-8'):

    try:
        with open(ulaz, encoding=kodiranje) as udat, \
             open(izlaz, 'w', encoding=kodiranje) as idat:

            for red in udat:
                idat.write(red)
            print('{} -> {}  OK!'.format(ulaz, izlaz))

    except UnicodeDecodeError:
        print(kodiranje, 'pogrešno kodiranje!')
    except FileNotFoundError:
        print('Pogrešna putanja!')
    except:
        print('Greška pri kopiranju!')
# test
kopiraj('c:\\papers\\test.txt','c:\\papers\\test2.txt')
kopiraj('c:\\papers\\test.txt','c:\\papers\\test2.txt','cp1252')
```

Nastavak programske linije