FUNKCIONALNO PROGRAMIRANJE

Oznaka predmeta: FPR

Predavanje broj: 04

Nastavna jedinica: PYTHON,

Nastavne teme:

Datum i vreme (struct_time, time, localtime, asctime, clock, ctime, gmtime, sleep, calendar, altzone, strftime, strptime, tzset). Atributi time-a. Funkcije modula calendar (firstdayweek, calendar, month, isleap, leapdays, monthrange, prcal, prmonth, setfirstweekday, timegm, weekday). Input, eval. Fajlovi (open, modovi, buffering, close, write, read, seek). Metode modula os (rename, remove, mkdir, rmdir, getcwd, chdir). Metode objekta file.

Predavač: prof. dr Perica S. Štrbac, dipl. ing.

Literatura:

Steven Lott: "Functional Python Programming", Packt Publishing, 2015.

Python: datum i vreme

- U Python-u funkcija *time.time()* vraća tekuće sistemsko vreme koje se računa od 1.1.1970. godine.

```
import time; # potrebno je ukljuciti ovaj modul
ticks = time.time()
print ("Number of ticks since 12:00am, January 1, 1970:", ticks)
```

```
Number of ticks since 12:00am, January 1, 1970: 1445942131.585914
```

- Vremenska n-torka

Indeks	Polje	Vrednost
0	4-digit year	Npr. 2008
1	Month	1 to 12
2	Day	1 to 31
3	Hour	0 to 23
4	Minute	0 to 59
5	Second	0 to 59
6	Day of Week	0 to 6 (0 is Monday)
7	Day of year	1 to 366 (Julian day)
8	Daylight savings	-1, 0, 1, -1 means library determines DST

Python: datum i vreme, struct_time

- Prethodna 9-torka je ekvivalentna strukturi **struct_time:**

Indeks	Atributi	Vrednosti
0	tm_year	2008
1	tm_mon	1 to 12
2	tm_mday	1 to 31
3	tm_hour	0 to 23
4	tm_min	0 to 59
5	tm_sec	0 to 61 (60 or 61 are leap-seconds)
6	tm_wday	0 to 6 (0 is Monday)
7	tm_yday	1 to 366 (Julian day)
8	tm_isdst	-1, 0, 1, -1 means library determines DST

- Translacija iz vrednosti datih u sekudama proteklim od 1.1.1970. godine u vremensku 9-torku je kao što sledi:

```
import time; localtime = time.localtime(time.time())
print ("Local current time :", localtime)
```

```
Local current time : time.struct_time(tm_year=2015, tm_mon=10, tm_mday=27,
tm_hour=11, tm_min=41, tm_sec=21, tm_wday=1, tm_yday=300, tm_isdst=0)
```

Python: datum i vreme, localtime, calendar, altzone

- Dohvatanje formatiranog vremena:

```
import time
asctime = time.asctime( time.localtime(time.time()) )
print ("Local current time :", asctime )
```

```
Local current time : Tue Jan 13 10:17:09 2009
```

- Prikaz kalendara za dati mesec

```
import calendar
calm = calendar.month(2015, 7)
print ("Here is the calendar:"); print (calm);
```

```
Here is the calendar:
     July 2015
Mo Tu We Th Fr Sa Su
       1  2  3  4  5
 6  7  8  9 10 11 12
13 14 15 16 17 18 19
20 21 22 23 24 25 26
27 28 29 30 31
```

Funkcije i varijable modula time:

- time.altzone - ofset lokalne DST vremenske zone u sekundama zapadno od UTC

```
import time
print ("time.altzone %d " % time.altzone)
```

```
time.altzone -7200
```

Python: datum i vreme,

- time.asctime([tupletime])

```
import time
t = time.localtime()
print ("time.asctime(t): %s " % time.asctime(t))
```

```
time.asctime(t): Thu Jul 16 14:47:44 2015
```

- time.clock( )

```
import time
def procedure():
    time.sleep(2.5)
# measure process time
t0 = time.clock()
procedure()
print (time.clock() - t0, "seconds process time")
# measure wall time
t0 = time.time()
procedure()
print (time.time() - t0, "seconds wall time")
```

```
2.5001568558090175 seconds process time
2.500142812728882 seconds wall time
```

- time.ctime([secs])

```
import time
print ("time.ctime() : %s" % time.ctime())
```

```
time.ctime() : Tue Feb 17 10:00:18 2009
```

Python, datum i vreme

- time.gmtime([secs])

```
import time
print ("time.gmtime() : %s" % time.gmtime())
```

```
time.gmtime() : time.struct_time(tm_year=2015, tm_mon=10, tm_mday=27,
tm_hour=11, tm_min=9, tm_sec=42, tm_wday=1, tm_yday=300, tm_isdst=0)
```

- time.localtime([secs])

```
import time
print ("time.localtime() : %s" % time.localtime())
```

```
time.localtime() : time.struct_time(tm_year=2015, tm_mon=10, tm_mday=27,
tm_hour=12, tm_min=12, tm_sec=29, tm_wday=1, tm_yday=300, tm_isdst=0)
```

- time.mktime(tupletime)

```
import time
t = (2009, 2, 17, 17, 3, 38, 1, 48, 0)
secs = time.mktime( t )
print ("time.mktime(t) : %f" % secs)
print ("asctime(localtime(secs)): %s" % time.asctime(time.localtime(secs)))
```

```
time.mktime(t) : 1234915418.000000
asctime(localtime(secs)): Tue Feb 17 17:03:38 2009
```

- time.sleep(secs)

```
import time
print ("S: %s" % time.ctime()); time.sleep(5)
print ("E: %s" % time.ctime())
```

```
S: Thu Jul 16 15:31:10 2015
E: Thu Jul 16 15:31:15 2015
```

Python, datum i vreme

- time.strftime(fmt[,tupletime])
  - formatira ispis datuma i vremena

```
import time
t = (2009, 2, 17, 17, 3, 38, 1, 48, 0)
t = time.mktime(t)
print (time.strftime("%b %d %Y %H:%M:%S", time.gmtime(t)))
                              Feb 18 2009 00:03:38
```

- time.strptime(str,fmt='%a %b %d %H:%M:%S %Y')

```
import time
struct_time = time.strptime("30 Nov 00", "%d %b %y")
print ("returned tuple: %s " % struct_time)
        returned tuple: (2000, 11, 30, 0, 0, 0, 3, 335, -1)
```

- time.time( )

```
import time
print ("time.time(): %f " % time.time())
print (time.localtime( time.time() ))
print (time.asctime( time.localtime(time.time()) ) )
```

```
time.time(): 1445946474.565745
time.struct_time(tm_year=2015, tm_mon=10, tm_mday=27, tm_hour=12, tm_min=47,
tm_sec=54, tm_wday=1, tm_yday=300, tm_isdst=0)
Tue Oct 27 12:47:54 2015
```

Python: popis formata za datum i vreme

- %a - abbreviated weekday name
- %A - full weekday name
- %b - abbreviated month name
- %B - full month name
- %c - preferred date and time representation
- %C - century number (the year divided by 100, range 00 to 99)
- %d - day of the month (01 to 31)
- %D - same as %m/%d/%y
- %e - day of the month (1 to 31)
- %g - like %G, but without the century
- %G - 4-digit year corresponding to the ISO week number (see %V).
- %h - same as %b
- %H - hour, using a 24-hour clock (00 to 23)
- %I - hour, using a 12-hour clock (01 to 12)
- %j - day of the year (001 to 366)
- %m - month (01 to 12)
- %M - minute
- %n - newline character
- %p - either am or pm according to the given time value
- %r - time in a.m. and p.m. notation
- %R - time in 24 hour notation

Python: popis formata za datum i vreme

- %S - second
- %t - tab character
- %T - current time, equal to %H:%M:%S
- %u - weekday as a number (1 to 7), Monday=1.
  Warning: In Sun Solaris Sunday=1
- %U - week number of the current year, starting with the first Sunday as the first day of the first week
- %V - The ISO 8601 week number of the current year (01 to 53), where week 1 is the first week that has at least 4 days in the current year, and with Monday as the first day of the week
- %W - week number of the current year, starting with the first Monday as the first day of the first week
- %w - day of the week as a decimal, Sunday=0
- %x - preferred date representation without the time
- %X - preferred time representation without the date
- %y - year without a century (range 00 to 99)
- %Y - year including the century
- %Z or %z - time zone or name or abbreviation
- %% - a literal % character

Python, datum i vreme

Atributi time-a:

- `time.timezone` — atribut predstavlja ofset u sekundama lokalne vremenske zone (bez DST) u odnosu na UTC (>0 u Americi; <=0 u većini evropskih država, u Aziji i Africi).
- `time.tzname` — atribut predstavlja deo uređenog para koga čine lokalana vremenska zona i bez ili sa DST, respektivno.

Funkcije modula calendar:

- `calendar.calendar(year,w=2,l=1,c=6)`

  Vraća mulitlinijski string koji predstavlja formatirani kalendar navedene godine (year). Meseci su raspoređeni u 3 kolone odvojene sa c razmaka. Parametar w određuje širinu datuma u karakterima. Svaka linija je duga 21*w+18+2*c. Parametar l predstavlja broj linija za sedmicu.
- `calendar.firstweekday()`

  Vraća tekuću postavku startnog dana u sedmici. Podrazumevano je 0 što znači (Ponedeljak).
- `calendar.isleap(year)`

  Vraća odgovor da li je godina (year) prestupna (True – ako jeste, inače False).

Python, datum i vreme

- `calendar.leapdays(y1,y2)`

  Vraća ukupan broj prestupnih dana u opsegu dve navedene godine (y1, y2).

- `calendar.month(year,month,w=2,l=1)`

  Vraća višelinijski string koji predstavlja kalendar navedenog meseca u navedenoj godini. Sedmica je u jednoj liniji ispisa. Zaglavlje ima dve linije (nayiv ima dve linije (mesec, dani u sedmici). Parametar w se odnosi na širinu svakog datuma u karakterima. Svaka linija ima dužinu 7*w+6. Parametar l se donosi na broj linija ispisa za svaku sedmicu.

- `calendar.monthcalendar(year,month)`

  Vraća lisu lista celih brojeva. Podlista predstavlja sedmicu. Dani van meseca imaju vrednost 0 a pripadani dani imaju dodeljen redni broj tog dana u mesecu.

- `calendar.monthrange(year,month)`

  Vraća uređen par celih brojeva. Prvi broj pokazuje indeks prvog dana u mesecu u podlisti sedmice. Drugi broj pokazuje koliko navedeni mesec ima dana.

- `calendar.prcal(year,w=2,l=1,c=6)`

  Kao print(calendar.calendar(year,w,l,c))

Python, datum i vreme

- `calendar.prmonth(year,month,w=2,l=1)`

  Kao print(calendar.month(year,month,w,l))

- `calendar.setfirstweekday(weekday)`

  Postavljanje koda prvog dana u sedmici. Kodovi su 0 (Monday) do 6 (Sunday).

- `calendar.timegm(tupletime)`

  Suprotno od time.gmtime. Prihvata vremensku 9-torku, a vraća broj sekundi koji je protekao od 1.1.1970. godine.

- `calendar.weekday(year,month,day)`

  Vraća kod dana u sedmici za dati datum. Kodovi su 0 (Monday) do 6 (Sunday). Meseci idu od 1 do 12.

Python, I/O

- Osnovna naredba za ispis na ekran je print:

```
print ("Hello, Python!");
print ("Python is really a great language,", "isn't it?");
```

```
Hello, Python!
Python is really a great language, isn't it?
```

- Python-ove ugrađene naredbe za prihvatanje sa standardnog ulaza:
  - input([prompt])
    - funkcija čita liniju sa standardnog ulaza i vraća je kao string bez oznake za novi red na kraju.

```
str = input("Enter your input: ");
print ("Received input is : ", str)
```

```
Enter your input: Hello Python
Received input is : Hello Python
```

    - za evaluaciju ulaza koristiti funkciju eval:

```
str = input("Unesi izraz: ");
lista = eval(str);
print (lista)
```

```
Unesi izraz: [x*5 for x in range(2,10,2)]
[10, 20, 30, 40]
```

Python, I/O

- Python obezbeđuje osnovne funkcije za manipulaciju fajlovima. Koristi se fajl objekat.
- Ugrađena funkcija open služi za otvaranje fajla pre njegove manipulacije.
  - Ova funkcija kreira fajl objekat a onda se dalje manipulacije fajlom sprovode pozivom odgovarajućuh metoda ovog objekta.

```
fd = open(file_name [, access_mode][, buffering])
```

  - file_name: predstavlja string koji sadrži ime fajla kome se želi pristupiti.
  - access_mode: određuje mod u kom će fajl biti otvoren (read, write, append, ...). Ovo je opcionalni parametar a njegova podrazumevana vrednost je read (r).
  - buffering:
    - ako je vrednost za baferovanje postavljena na 0 nema baferovanja fajla.
    - ako je vrednost baferovanja postavljena na 1, izvršava se line buffering pri pristupanju fajlu.
    - Ako je vrednost >1 onda ona znači veličinu bafera. Ako je vrednost negativna onda je veličina bafera na podrazumevanoj sistemskoj vrednosti.

Python, I/O, modovi otvaranja fajla

Mod	Opsi
r	Čitanje. File pointer gleda na početak fajla. Podrazumevani mod.
rb	Čitanje u binarnom formatu. File pointer gleda na početak fajla.
r+	Čitanje i pisanje. File pointer gleda na početak fajla.
rb+	Čitanje i pisanje u binarnom formatu. File pointer gleda na početak fajla.
w	Pisanje. Obrisaće postojeći fajl, odnosno, kreiraće novi fajl za pisanje ako takav fajl ne postoji.
wb	Pisanje u binarnom formatu. Obrisaće postojeći fajl, odnosno, kreiraće novi fajl za pisanje u binarnom formatu ako takav fajl ne postoji.
w+	Čitanje i pisanje. Obrisaće postojeći fajl, odnosno, kreiraće novi fajl za čitanje i pisanje ako takav fajl ne postoji.
wb+	Čitanje i pisanje u binarnom formatu. Obrisaće postojeći fajl, odnosno, kreiraće novi fajl za čitanje i pisanje u binarnom formatuako takav fajl ne postoji.
a	Dodavanje. File pointer gleda na kraj postojećeg fajla, odnosno, kreira se novi fajl za pisanje ako takav ne postoji.

Python, I/O, modovi otvaranja fajla, atributi file object-a

Mod	Opsi
ab	Dodavanje u binarnom formatu. File pointer gleda na kraj postojećeg fajla, odnosno, kreira se novi fajl za pisanje u binarnom formatu ako takav ne postoji.
a+	Dodavanje i čitanje. File pointer gleda na kraj postojećeg fajla, odnosno, kreira se novi fajl za čitanje i pisanje ako takav ne postoji.
ab+	Dodavanje i čitanje u binarnom formatu. File pointer gleda na kraj postojećeg fajla, odnosno, kreira se novi fajl za čitanje i pisanje ako takav ne postoji.

- Atributi *file* object-a

Atribute	Opis
file.closed	Vraća true ako je fajl zatvoren, inače vraća false.
file.mode	Vraća mod pristupa u kom je fajl otvoren.
file.name	Vraća ime fajla.

Python, I/O

- Primer:

```
fo = open("foo.txt", "wb")
print ("Name of the file: ", fo.name)
print ("Closed : ", fo.closed)
print ("Opening mode : ", fo.mode)
```

```
Name of the file: foo.txt
Closed : False
Opening mode : wb
```

- Metod `close()` file object-a zatvara fajl.
  - Python automatski zatvara fajl ako se referencirani objekt dodeli drugom fajlu.
  - Bolje je koristiti eksplicitno metod `close()`.

```
fileObject.close();
```

Primer:

```
fo = open("foo.txt", "wb")
print ("Name of the file: ", fo.name)
fo.close()
print(fo.closed)
```

```
Name of the file: foo.txt
True
```

Python, I/O

- Za čitanje i pisanje fajlova koriste se metode:
  - write()
    - write metod upisuje string u otvoreni fajl.
    - write metod ne dodaje znak za novi red ('\n') na kraj stringa

```
fileObject.write(string);
```

primer:

```
fo = open("foo.txt", "wb")
fo.write(b"Python is a great language.\nYeah its great!!\n");
fo.close()
```

```
Python is a great language. Yeah its great!!
```

  - read()
    - metod read čita string iz otvorenog fajla.

```
fileObject.read([count]);
```

prosleđeni parametar je broj bajtova koji će biti pročitani iz otvorenog fajla počevši od početka fajla a ako parametar count nije naveden pokušava se čitanje što više podatka (možda do kraja fajla).

```
fo = open("foo.txt", "r+")
str = fo.read(10);
print ("Read String is : ", str)
fo.close()
```

```
Read String is : Python is
```

Python, I/O

- Pozicioniranje unutar fajla:
  - *tell()* metod vraća tekuću poziciju u fajlu.
  - *seek(offset[, from])* metod menja tekuću fajl poziciju.
    - *offset* argument označava pomeraj pozicije u fajlu.
    - *from* argument označava odakle se računa pomeraj.

      0 pomeraj se računa od početka fajla
      1 pomeraj je od tekuće pozicije u fajlu
      2 pomeraj se računa od kraja fajla

      (f.seek(-3, 2)).

```
fo = open("foo.txt", "r+")
str = fo.read(10);
print ("Read String is : ", str)
position = fo.tell();
print ("Current file position : ", position)
position = fo.seek(0, 0);
str = fo.read(10);
print ("Again read String is : ", str)
fo.close()
```

```
Read String is : Python is
Current file position : 10
Again read String is : Python is
```

Python, I/O

- Python-ov modul **os** obezbeđuje metode za procesiranje operacija nad fajlovima.
  - Za korišćenje modula dovoljno je uraditi import ovog modula.
- Preimenovanje fajlova obavlja metoda rename().
- Metoda *rename()* ima 2 argumenta, tekući naziv fajla i novi naziv fajla.
  (os.rename(current_file_name, new_file_name)).

```
import os
# preimenovanje test1.txt u test2.txt
os.rename( "test1.txt", "test2.txt" )
```

- Brisanje fajlova obavlja metoda remove() kojoj se prosleđuje ime fajla koji se briše (`os.remove(file_name)`).

```
import os
os.remove("text2.txt")
```

- Modul **os** ima nekoliko metoda za kreiranje, uklanjanje i menjanje direktorijuma.
- Metod *`mkdir()`* kreira u tekućem direktorijumu direktorijum čiji je naziv prosleđen kao argument.

```
os.mkdir("newdir")
```

Primer:

```
import os
os.mkdir("test")
```

Python, I/O

- Metod *chdir()* menja tekući direktorijum, jedini argument je naziv novog tekućeg direktorijuma.

```
os.chdir("newdir")
```

Primer:

```
import os
os.chdir("/home/newdir")
```

- Metod *getcwd()* ispisuje tekući radni direktorijum.

```
os.getcwd()
```

Primer:

```
import os
os.getcwd()
```

- Metod *rmdir()* uklanja direktorijum čiji je naziv prosleđen ako parametar.
  - Pre uklanjanja direktorijuma, ceo sadržaj unutar direktorijuma mora biti uklonjen.

```
os.rmdir('dirname')
```

Primer:

```
import os
os.rmdir( "/tmp/test" )  # ako je /tmp/test prazan
```

Python: I/O, neke metode file object-a

- `file.close()`, zatvara fajl.
- `file.flush()`, prazni interni bafer (kao fflush kod stdio)
- `file.fileno()`, vraća celobrojnu vrednost file descriptor-a.
- `file.read([size])`, čita najviše *size* bajtova fajla (manje ako ranije dosegne EOF).
- `file.readline([size])`, čita unutar linije fajla (uzeće i završni '\n').
- `file.readlines([sizehint])`, vraća listu linija fajla. Opcionalni parametar sizehint označava koliko će se ukupno bajtova pročitati računajući sve linije.
- `file.seek(offset[, whence])`, postavljanje tekuće pozicije u fajlu.
- `file.tell()`, vraća tekuću poziciju u fajlu.
- `file.truncate([size])`, odseca fajl do na maksimalno datu dužinu.
- `file.write(str)`, upisuje string u fajl (vraća dužinu upisa).
- `file.writelines(sequence)`, upisuje sekvencu stringova u fajl.

Python, metode file object-a

```
fo = open("foo.txt", "wb")
print ("Name of the file: ", fo.name)
fo.close()
```

```
Name of the file: foo.txt
```

```
fo = open("foo.txt", "wb")
print ("Name of the file: ", fo.name)
fo.flush()
fo.close()
```

```
Name of the file: foo.txt
```

```
fo = open("foo.txt", "wb")
print ("Name of the file: ", fo.name)
fid = fo.fileno()
print ("File Descriptor: ", fid)
fo.close()
```

```
Name of the file: foo.txt
File Descriptor: 3
```

Python, metode file object-a

```
fo = open("foo.txt", "r")
print ("Name of the file: ", fo.name)
for index in range(5):
   line = fo.readline()
   print ("Line No %d - %s" % (index, line))
fo.close()
```

```
Name of the file: foo.txt
Line No 0 - This is 1st line

Line No 1 - This is 2nd line

...sve do 5-te linije sa uračunatim \n
```

```
# sadrzaj foo.txt fajla
za 4 primera koji slede:
This is 1st line
This is 2nd line
This is 3rd line
This is 4th line
This is 5th line
```

```
fo = open("foo.txt", "r")
line = fo.read(10)
print ("Read Line: %s" % (line))
fo.close()
```

```
Read Line: This is 1s
```

```
fo = open("foo.txt", "r")
line = fo.readline()
print ("Read Line: %s" % (line))
line = fo.readline(5)
print ("Read Line: %s" % (line))
fo.close()
```

```
Read Line: This is 1st line
Read Line: This
```

Python, metode file object-a

```
fo = open("foo.txt", "r")
line = fo.readlines()
print ("Read Line: %s" % (line))
line = fo.readlines(2)
print ("Read Line: %s" % (line))
fo.close()

Read Line: ['This is 1st line\n', 'This is 2nd line\n', 'This is 3rd
line\n', 'This is 4th line\n', 'This is 5th line\n']
Read Line: []
```

•Metod seek() postavlja tekuću poziciju u fajlu u odnosu na početak, tekuću vrednost ili kraj fajla (0, 1 i 2 respektivno).

- Nema povratne vrednosti.
- Ako je fajl otvoren za dodavanje korišćenjem 'a' or 'a+', bilo koja seek() operacija biće poništena pri sledećem upisu.
  - Ako je fajl otvoren za pisanje u modu dodavanja 'a' seek() nema uticaja osim ako je u pitanju operacija čitanja (mode 'a+').

Python, I/O

```
fo = open("foo.txt", "r")
print ("Name of the file: ", fo.name)
line = fo.readline()
print ("Read Line: %s" % (line))
# Again set the pointer to the beginning
fo.seek(0, 0)
line = fo.readline()
print ("Read Line: %s" % (line))
# Close opend file
fo.close()
```

```
Name of the file: foo.txt
Read Line: This is 1st line
Read Line: This is 1st line
```

```
fo = open("foo.txt", "r")
print ("Name of the file: ", fo.name)
line = fo.readline()
print ("Read Line: %s" % (line))
# Get the current position of the file.
pos = fo.tell()
print ("Current Position: %d" % (pos))
fo.close()
```

```
Name of the file: foo.txt
Read Line: This is 1st line
Current Position: 18
```

Python, I/O

```
fo = open("foo.txt", "r+")
print ("Name of the file: ", fo.name)
line = fo.readline()
print ("Read Line: %s" % (line))
# Now truncate remaining file.
fo.truncate()
# Try to read file now
line = fo.readline()
print ("Read Line: %s" % (line))
# Close opend file
fo.close()
```

```
Name of the file: foo.txt
Read Line: This is 1st line
Read Line:
```

Python, I/O

```
fo = open("foo.txt", "r+")
print ("Name of the file: ", fo.name)
str = "This is 6th line"
# Write a line at the end of the file.
fo.seek(0, 2)
line = fo.write( str )

# Now read complete file from beginning.
fo.seek(0,0)
for index in range(6): # napisite resenje koje nije hard-kodovano
   line = fo.readline()
   print ("Line No %d - %s" % (index, line))
# Close opend file
fo.close()
```

```
Name of the file: foo.txt
Line No 0 - This is 1st line
Line No 1 - This is 2nd line
Line No 2 - This is 3rd line
Line No 3 - This is 4th line
Line No 4 - This is 5th line
Line No 5 - This is 6th line
```

Python, I/O

```
fo = open("foo.txt", "r+")
print ("Name of the file: ", fo.name)
seq = ["This is 6th line\n", "This is 7th line"]

# Write sequence of lines at the end of the file.
fo.seek(0, 2)
line = fo.writelines( seq )
# Now read complete file from beginning.
fo.seek(0,0)
for index in range(7): # napisite resenje koje nije hard-kodovano
   line = fo.readline()
   print ("Line No %d - %s" % (index, line))
# Close opend file
fo.close()
```

```
Name of the file: foo.txt
Line No 0 - This is 1st line
Line No 1 - This is 2nd line
Line No 2 - This is 3rd line
Line No 3 - This is 4th line
Line No 4 - This is 5th line
Line No 5 - This is 6th line
Line No 6 - This is 7th line
```