# Fișiere în Python

## 1. Utilizare *open( ) ... close( )*

### Citirea unui fisier text linie cu linie

```
f = open('note_stud.txt', 'r')
while True:
    linie = f.readline()
    if len(linie)==0:
        break
    print(linie, end='')
f.close()
```

```
=== RESTART: ….. ===========
Georgescu Ion 9
Petrescu Petre 8
Zamfirescu Zamfira 10
```

**Fisierul note_stud.txt**

```
Georgescu Ion 9
Petrescu Petre 8
Zamfirescu Zamfira 10
```

### O alta varianta – lista

```
f = open('note_stud.txt', 'r')
lst = list(f)
print(lst)
# . . .
print(f)

f.close()
```

```
 ['Georgescu Ion 9\n', 'Petrescu
Petre 8\n', 'Zamfirescu Zamfira
10\n']
. . .
<_io.TextIOWrapper
name='note_stud.txt' mode='r'
encoding='cp1252'>
```

Afisarea numarului inregistrarilor

```
f = open('note_stud.txt', 'r')
lst = list(f)
f.close()

for i in range (len(lst)):
    print( i+1,':\t',lst[i])
```

```
1 :    Georgescu Ion 9

2 :    Petrescu Petre 8

3 :    Zamfirescu Zamfira 10
```

### Enumerare

```
f = open('note_stud.txt', 'r')
lst = list(f)
f.close()

for i, linie in enumerate(lst):
    print( i+1, ':\t', linie, end='')
```

```
1 :    Georgescu Ion 9
2 :    Petrescu Petre 8
3 :    Zamfirescu Zamfira 10
>>>
```

# 2.Utilizare *with open( ) as …*

## with open ( nume_fisier, mod ) as f:

**Functia close() e apelata automat cand se iese din blocul with**

```
with open('note_stud.txt', 'rt') as f:
    linii=f.readlines()

print(linii)
print()
for linie in linii:
    print(linie.rstrip())
    #nu e necesar end=''
```

```
['Georgescu Ion 9\n',
'Petrescu Petre 8\n',
'Zamfirescu Zamfira 10\n']

Georgescu Ion 9
Petrescu Petre 8
Zamfirescu Zamfira 10
```

Acelasi rezultat si daca se utiliza `linii=list(f)`

Sau si mai concis, avand in vedere ca f este un iterabil

```
with open('note_stud.txt', 'rt') as f:
    for linie in f:
        print(linie.rstrip())
```

```
Georgescu Ion 9
Petrescu Petre 8
Zamfirescu Zamfira 10
```

## Citire campuri – split()

```
with open('note_stud.txt', 'rt') as f:
    linii=list(f)

n = media =0
print('Nume','\tPrenume','Nota',sep='\t')
for linie in linii:
    campuri = linie.split()
    nota = int(campuri[2])
    n, media = n+1, media+nota

print(campuri[0],campuri[1],nota,sep='\t')
print("Media celor",n,"studenti
este",media/float(n))
```

```
Nume        Prenume  Nota
Georgescu   Ion      9
Petrescu    Petre    8
Zamfirescu  Zamfira  10
Media celor 3 studenti este 9.0
>>>
```

## Prelucrare matrice campuri in stil C

```
with open('note_stud.txt', 'rt') as f:
    linii=f.readlines()

matcamp=[]
for linie in linii:
    matcamp.append(linie.split())

print(matcamp)
# prelucrare in stil C
for i in range(len(matcamp)):
    for j in range( len(matcamp[i]) ):
        print ( matcamp[i][j],
                        end='\t')
    print()
```

```
[['Georgescu', 'Ion',
'9'], ['Petrescu',
'Petre', '8'],
['Zamfirescu', 'Zamfira',
'10']]

Georgescu  Ion   9
Petrescu   Petre 8
Zamfirescu Zamfira    10
>>>
```

# Citirea unei matrici dintr-un fisier

```
mat = []
with open('matrice.txt', 'rt') as f:
    for linie in f:
        mat.append (linie.split())

print(mat)
```

```
[['-1', '2', '-3', '4'],
['5', '-6', '7', '-8'],
['-9', '10', '-11', '12']]
```

**Fisierul `matrice.txt`**

```
-1 2 -3   4
5  -6 7  -8
-9 10 -11 12
```

**In programul anterior elementele sunt siruri** !

Pentru conversie se poate apela element cu element o functie de conversie: int(x), float(x), …

# Conversia campurilor unei matrici

Daca se doreste conversia lor la int

```
mat=[]
for linie in open('matrice.txt','rt'):
    campuri=[]
    for x in linie.split():
        campuri.append(int(x))
    mat.append( campuri )
print(mat)
```

```
[[-1, 2, -3, 4], [5, -6, 7, -8], [-9, 10, -11, 12]]
```

Sau mai concis folosind "list comprehension".

```
mat=[]
for linie in open('matrice.txt','rt',):
    campuri = [int(x) for x in linie.split()]
    mat.append( campuri )
print(mat)
```

```
[[-1, 2, -3, 4], [5, -6, 7, -8], [-9, 10, -11, 12]]
```

Se poate utiliza însă functia *map()*

## Functia builtin map()

**map( functie, iterabil ) -> iterabil**

```
def f(x):
    return x*x

for x in map(f, range(3)):
    print(x)

lst=[10,20, 30]
print(  list( map(f, lst)) )
```

```
0
1
4


[100, 400, 900]
>>>
```

## Conversie campuri cu map()

```
mat = []
with open('matrice.txt', 'rt') as f:
    for linie in f:
        campuri = map (int, linie.split())
        mat.append( list(campuri) )

print(mat)
```

```
[[-1, 2, -3, 4],
[5, -6, 7, -8],
[-9, 10, -11, 12]]
>>>
```

In **map()** s-a apelat functia **int(x)**

**Fisierul `matrice.txt`**

```
-1 2 -3   4
5  -6 7  -8
-9 10 -11 12
```

Ar trebui realizat un modul care sa contina o functie de citire a unei matrici din fisier.

## Afisarea elementelor maxime pe linii

elemMaxime.py

```
import citMatFis as c

w = c.citesteMatrice('matrice.txt')

for linie in w:
    print(linie,'max=',max(linie))
```

citMatFis.py

```
def citesteMatrice(numeFisier, mod='rt',
tip=int):
    mat = []
    with open(numeFisier, mod) as f:
        for linie in f:
            linconv = list( map(tip,
                                linie.split()))
            mat.append (linconv)

    return mat

if __name__ == '__main__':
    print( citesteMatrice('matrice.txt') )
    print('Matricea float')
    print( citesteMatrice('matrice.txt',
                          'rt', float) )
```

```
[-1, 2, -3, 4] max= 4
[5, -6, 7, -8] max= 7
[-9, 10, -11, 12] max= 12
>>>
```

```
[[-1, 2, -3, 4], [5, -6, 7, -8], [-9, 10, -11,
12]]
Matricea float
[[-1.0, 2.0, -3.0, 4.0], [5.0, -6.0, 7.0, -
8.0], [-9.0, 10.0, -11.0, 12.0]]
>>>
```

De observat ca `citesteMatrice()` primeste din al doilea apel ca argument o functie, *float*, pe care o transmite mai departe.

***Exercitii***. 1. Sa se determine maximele pe coloane si elementul maxim din matrice.

2. Sa se scrie o functie de citire a elementelor unui vector indiferent cum sunt dispuse in fisier.

# Scrierea unui fișier text

Se deschide cu 'wt' sau 'at' si se scrie cu functia `f.write(sir)`

```
import random as r
import citMatFis as c
with open('matw.txt', 'wt') as f:
    for i in range(4):
        linie=""
        for j in range(5):
            linie += str( r.randint(5,10) ) +' '
        f.write(linie)
        f.write('\n')

print (c.citesteMatrice('matw.txt'))
```

```
[[6, 6, 10, 6, 9],
[10, 6, 6, 10, 9],
[10, 9, 9, 7, 10],
[5, 9, 10, 6, 7]]
```

Se reminteste ca dupa **with** nu e necesar **close()**.

In Python se pot prelucra si fisiere binare…

## PICKLE

Este posibila salvarea si restaurarea obiectelor (lista, multime, …) in fisiere in format binar cu ajutorul modulului Pickle

```
import pickle

lst1 = [2*k+1 for k in range(10)]
with open('lista_1.pkl', 'wb') as f:
    pickle.dump(lst1, f)

with open('lista_1.pkl', 'rb') as f:
    impare = pickle.load ( f )

print(lst1)
print(impare)
```