Εισαγωγή στον προγραμματισμό με τη γλώσσα Python

Η γλώσσα Python

- Ελεύθερη, ηλεκτρονικά διαθέσιμη (www.python.org)
- Κατάλληλη για διδασκαλία προγρ/σμού
- Χρησιμοποιεί διερμηνευτή
- Υποστηρίζει διαδικασιακό, συναρτησιακό και αντικειμενοστραφή προγραμματισμό
- Επαγγελματική χρήση
- Windows, Linux, Mac

Η Python στην εκπαίδευση

- Βασικός σκοπός είναι η μεταφορά της αλγοριθμικής λύσης ενός προβλήματος σε ένα σύγχρονο και ευρέως γνωστό προγραμματιστικό περιβάλλον, δηλαδή η προσέγγιση των βασικών αρχών αλγοριθμικής και προγραμματισμού με εργαλείο την Python.
- Δεν έχει σκοπό την σε βάθος εκμάθηση της γλώσσας (βιβλιοθήκες, δυνατότητες νέων εκδόσεων κλπ).

Γιατί Python;

- Απλή και ευέλικτη σύνταξη
- Άμεση ανατροφοδότηση (ως διερμηνευόμενη γλώσσα)
- Ελεύθερη
- Υποστήριξη από κοινότητα ΕΛ/ΛΑΚ προγραμματιστών
- Βιβλιοθήκες (πχ turtle – Logo)
- Ευρεία διάδοση στις Πανεπιστημιακές σχολές

Παιδαγωγική αξιοποίηση

Εργαστηριακή προσέγγιση: οι μαθητές μαθαίνουν βασικές έννοιες αλληλεπιδρώντας με το διερμηνευτή της Python (διερευνητική προσέγγιση που βασίζεται στην εποικοδομιστική μάθηση).

Δραστηριότητες

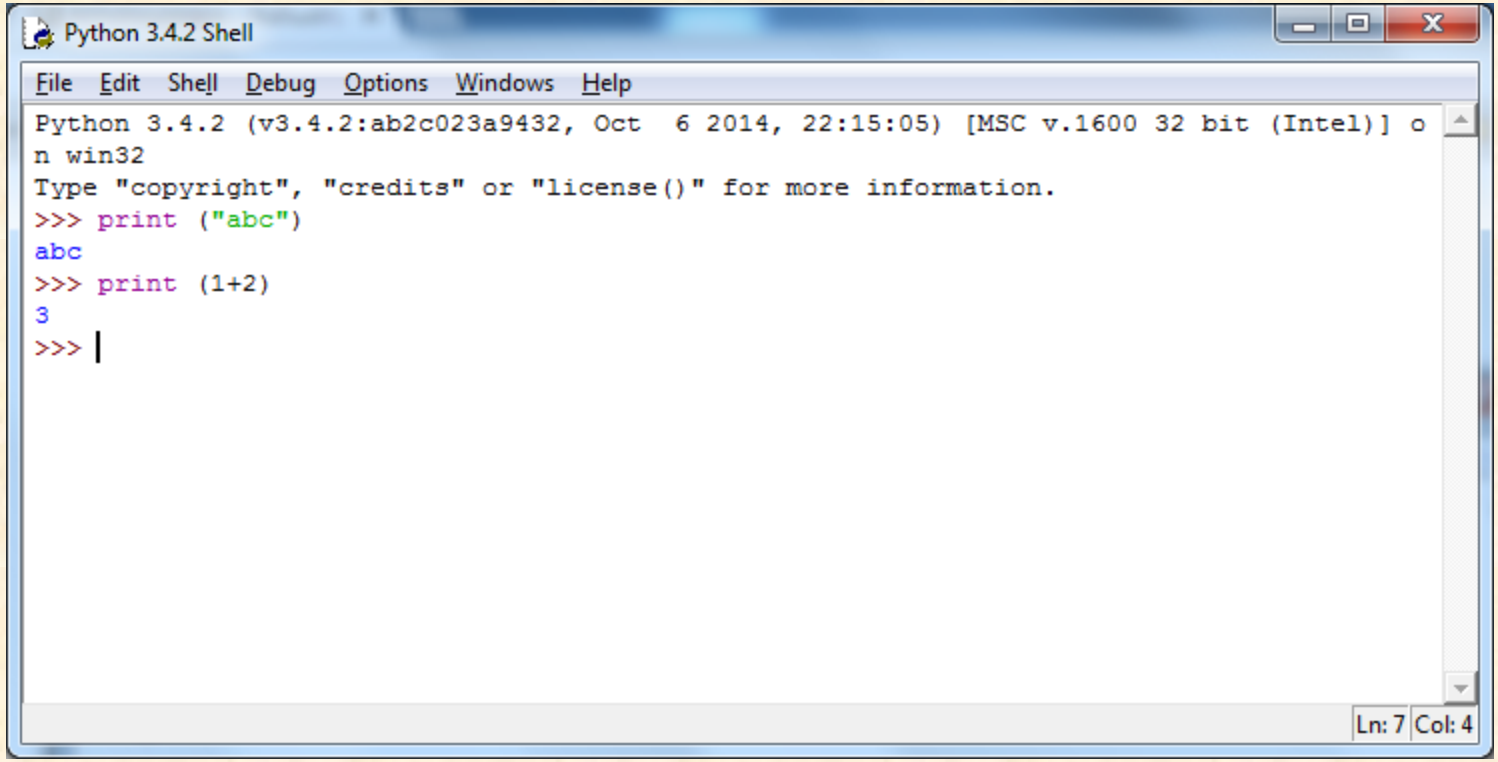
- Τι κάνει ένα έτοιμο (απλό) πρόγραμμα. Μελέτη του κώδικα
- Συμπλήρωση ημιτελούς προγράμματος
- Διόρθωση – προσαρμογή έτοιμου προγράμματος
- Δημιουργία ολόκληρου προγράμματος για την επίλυση συγκεκριμένου προβλήματος
- Μικρά projects

Ορισμένα χαρακτηριστικά

Τα block εντολών που περιέχονται σε εντολές επιλογής, επανάληψης κλπ καθορίζονται από τη στοίχιση (**εσοχή**) και μόνο:

```
If x > 0 :
    print “OK”
else:
    print “Error”
    print “Try again”
print “continue”
```

Ολοκληρωμένο περιβάλλον ανάπτυξης (IDLE)



Τύποι δεδομένων

- Αριθμοί
  1. Ακέραιοι (int): **2, -32**
  2. Κινητής υποδιαστολής (float): **2.3, 4.5e+20**
  3. Μιγαδικοί (complex): **(-2+3j)**
- Λογικοί (bool): **True, False**
- Συμβολοσειρές (str): **‘abc’, “ΑΒΓΔ xyz”**

Μεταβλητές

- Δεν απαιτείται δήλωση των μεταβλητών
- Ο τύπος μιας μεταβλητής καθορίζεται από την τιμή που παίρνει και μπορεί να αλλάζει.
- Εντολή εκχώρησης: X=1

```
>>> a=10
>>> type(a)
<class 'int'>
>>> a='abc'
>>> type(a)
<class 'str'>
>>>
```

*** Η συνάρτηση type() επιστρέφει τον τύπο του ορίσματος**

Μεταβλητές

Κανόνες ονομάτων

- Αποτελούνται από γράμματα, αριθμούς και _ (underscore)
- Αρχίζουν με γράμμα ή _
- Διάκριση πεζών – κεφαλαίων
- Μπορούν να χρησιμοποιηθούν ελληνικά γράμματα

Τελεστές

Τελεστής	Σημασία	Παραδείγματα
+	Πρόσθεση αριθμών ή ένωση string	1+2 →3 ‘a’ + ‘b’ → ‘ab’
-	Αφαίρεση	
*	Πολλαπλασιασμός	
**	Ύψωση σε δύναμη	2 ** 4 →16
/	Διαίρεση (αποτέλεσμα δεκαδικό)	5/2 → 2.5
//	Ακέραια διαίρεση (στρογγ. κάτω)	5//2 → 2
%	Υπόλοιπο ακέραιας διαίρεσης	5%2 → 1

Εντολή εκχώρησης

```
a = 10
b = b + 1
X = ‘Python’
a = b = c = 100
x, y = 10, 20
Z = True
T = 3 * ‘abc’
```

Είσοδος δεδομένων

Η συνάρτηση **input()**

x=input()

y=input (‘Δώστε την τιμή του y: ’)

Message=input(‘Enter your name:’)

* Η συνάρτηση **input()** επιστρέφει string.

```
>>> x=input()
5
>>> print(x)
5
>>> x
'5'
>>> type(x)
<class 'str'>
>>>
```

Συναρτήσεις μετατροπής τύπων

int(x): μετατρέπει κινητής υποδιαστολής ή string σε ακέραιους

float(x): μετατρέπει ακέραιους ή string σε κινητής υποδιαστολής

str(n): μετατρέπει αριθμούς σε string

```
>>> a=input()
5
>>> b=int(a)
>>> type(b)
<class 'int'>
>>> c=int(input())
7
>>> type(c)
<class 'int'>
>>>
```

Έξοδος αποτελεσμάτων

Συνάρτηση εξόδου **print()**

```
>>> x=10
>>> print(x)
10
>>> print(2+3)
5
>>> print(1>2)
False
>>>
```

- Στην Python 2.* δεν υπάρχει συνάρτηση print() αλλά εντολή print:

print 1+2

Λογικές εκφράσεις

Συγκριτικοί τελεστές

==

!=

>

<

>=

<=

Λογικοί τελεστές

not

and

or

Εντολή επιλογής if

```
if συνθήκη:
    Εντολές
```

```
if συνθήκη:
    Εντολές1
else:
    Εντολές2
```

```
if συνθήκη1:
    Εντολές1
elif συνθήκη2:
    Εντολές2
  . . .
else:
    ΕντολέςN
```

* Οι Εντολές πρέπει να είναι μετατοπισμένες κατά 4 κενά (εσοχή). Δεν υπάρχουν άλλες ενδείξεις αρχής- τέλους block εντολών (πχ {}, begin-end κλπ).

Παραδείγματα εντολής if

```
x=int(input('Δώστε ένα ακέραιο: '))
if x%2==0:
    print('ΑΡΤΙΟΣ')
else:
    print('ΠΕΡΙΤΤΟΣ')
print('ΤΕΛΟΣ')
```

Παραδείγματα εντολής if

```
x=float(input('Δώστε ένα αριθμό: '))
if x > 0:
    print('Θετικός')
elif x < 0:
    print('Αρνητικός')
else:
    print('Μηδέν')
print('ΤΕΛΟΣ')
```

Εμφωλευμένες if

```
x=float(input('Δώστε ένα αριθμό: '))
if x == 0:
    print('Μηδέν')
else
    if x < 0:
        print('Αρνητικός')
    else
        print('Θετικός')
   print('Μηδέν')
print('ΤΕΛΟΣ')
```

Εντολή επανάληψης for

```
for i in range(10):
    print(i)
```

```
0
1
2
3
4
5
6
7
8
9
```

Εντολή επανάληψης for

```
for i in range(5, 10):
    print(i)
```

5 6 7 8 9

```
for i in range(1, 20, 5):
    print(i)
```

1 6 11 16

```
for i in range(5, 1, -1):
    print(i)
```

5 4 3 2

```
for i in ('a','b','c'):
    print(i)
```

a b c

Εντολή επανάληψης while

```
i=1
while i<5:
    print(i)
    i=i+1
```

```
1
2
3
4
```

Ασκήσεις

Φύλλο Εργασίας 1

Άσκηση 1.2

Δημιουργήστε ένα πρόγραμμα Python το οποίο διαβάζει την ακτίνα ενός κύκλου και εμφανίζει το εμβαδόν του.

Λύση

```
r=float(input('Δώστε την ακτίνα: '))
pi=3.14
e=pi*r**2
print('Εμδαδόν = ',e)
```

Άσκηση 1.3

Γράψτε πρόγραμμα το οποίο διαβάζει δύο αριθμούς και εμφανίζει τον μεγαλύτερο, ή κατάλληλο μήνυμα αν είναι ίσοι.

Λύση

```
a=float(input('Δώστε το α: '))
b=float(input('Δώστε το β: '))
if a>b:
    print('Ο α μεγαλύτερος')
elif b>a:
    print('Ο β μεγαλύτερος')
else:
    print(‘Ισοι')
```

Άσκηση 1.4

Γράψτε πρόγραμμα που διαβάζει 10 αριθμούς και υπολογίζει και εμφανίζει το άθροισμά τους και το μέσο όρο τους.

Λύση

```
s=0
for i in range(10):
    a=float(input('Δώστε αριθμό: '))
    s=s+a
print('Άθροισμα = ',s)
print('Μέσος όρος = ',s/10)
```

Άσκηση 1.5

Γράψτε πρόγραμμα που διαβάζει αριθμούς μέχρι να διαβάσει το 0 και εμφανίζει το πλήθος των αρνητικών αριθμών που διάβασε.

Λύση

```
N=0
a=float(input('Δώστε αριθμό(0 για τέλος) :'))
while a!=0:
    if a<0:
        N=N+1
    a=float(input('Δώστε αριθμό(0 για τέλος) :'))
print('Πλήθος αρνητικών = ',N)
```

Συναρτήσεις

- Η Python διαθέτει ένα είδος υποπρογραμμάτων, τις **συναρτήσεις** οι οποίες μπορούν να εκτελούν όλες τις λειτουργίες.
- Δηλώνονται (συνήθως) στην αρχή του προγράμματος (και σίγουρα πριν χρησιμοποιηθούν)
- Μπορούν να επιστρέφουν τιμές ή όχι

Σύνταξη συνάρτησης

def Όνομα_συνάρτησης ([Παράμετροι]):

Εντολές

[return αποτέλεσμα]

Η κλήση της συνάρτησης γίνεται με αναφορά του ονόματός της (και των παραμέτρων, αν υπάρχουν)

print10()

ypol(a,b,c)

X=ypol(a,b,c)

Παραδείγματα συναρτήσεων

```
def print10():
    for i in range(1,11):
        print(i)

print10()
```

Δήλωση συνάρτησης

Η στοίχιση των εντολών καθορίζει το τέλος της συνάρτησης

Κλήση συνάρτησης

Συνάρτηση με παραμέτρους

```
def print_sum(a,b):
    print(a+b)

print_sum(1,2)
x=100
y=200
print_sum(x,y)
print_sum('ΚΑΛΗ','ΜΕΡΑ‘)
```

3

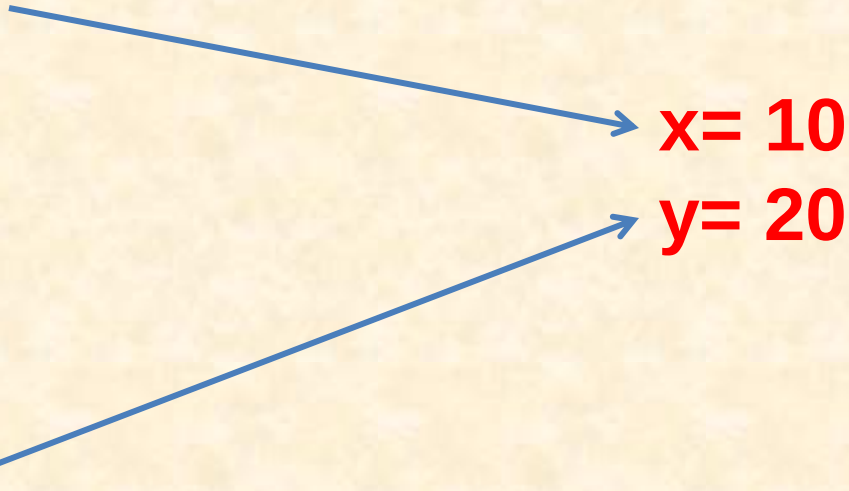
300

ΚΑΛΗΜΕΡΑ

Πέρασμα παραμέτρων με τιμή (by value)

```
def do10(x):
    x=10
    print('x=',x)

y=20
do10(y)
print('y=',y)
```



Συνάρτηση: επιστροφή τιμών
Η εντολή return

```
def mesos_oros(a,b):
    return (a+b)/2


print(mesos_oros(3,4))

z=mesos_oros(100,200)
print(z)
```

Συμβολοσειρές (strings)

Συμβολοσειρές (strings)

X = ‘abc’ + ‘def’ → **‘abcdef’**

X = 3 * ‘aaa’ → **‘aaaaaaaaa’**

'x' in 'abcdef‘ → False

‘c' in 'abcdef‘ → True

```
wrd = ‘ΤΑΞΗ’
for c in wrd
    print(c)
```

→

```
Τ
Α
Ξ
Η
```

Μέθοδοι Συμβολοσειρών

Παραδείγματα

s.isalpha()

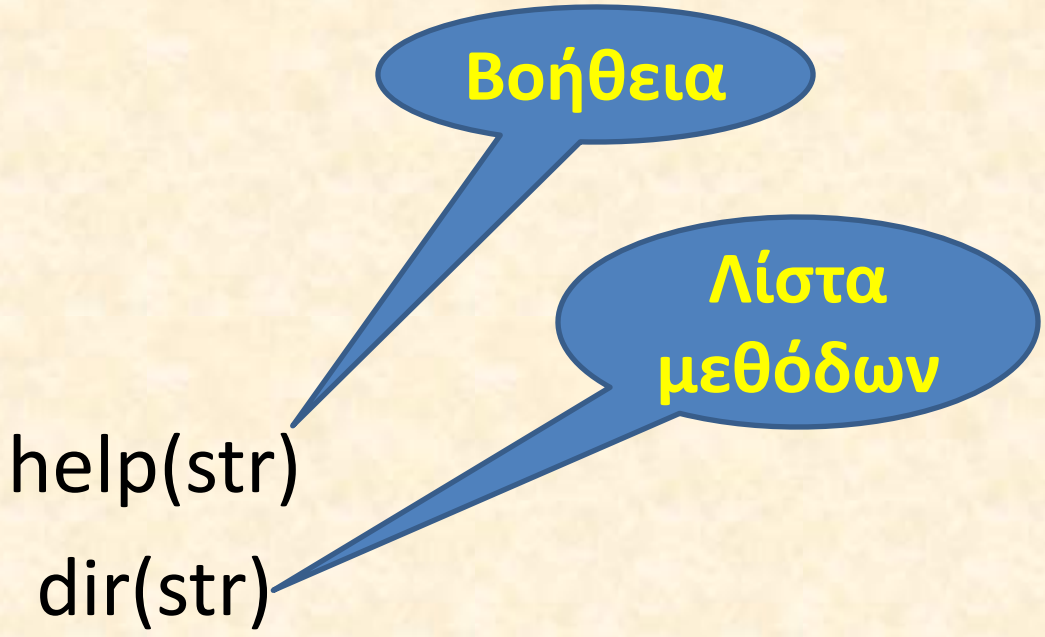
s.isdigit()

s.find('x')

s.count('a')

s.upper()

s.lower()



Η “πολυμορφική” συμπεριφορά της Python

```
def pollap(a, b):
    return(a*b)
```

```
print(pollap(4, 5))
```
→ 20

```
print(pollap('abc', 3))
```
→ abcabcabc

```
print(pollap('abc', 'xy'))
```
→ Run-time error

Lazy evaluation – αποφάσεις λαμβάνονται κατά το χρόνο εκτέλεσης

Οι συμβολοσειρές είναι αμετάβλητες (σε αντίθεση με τις λίστες)

```
>>> s='1234567'
>>> s[1]='a'
Traceback (most recent call last):
  File "<pyshell#1>", line 1, in <module>
    s[1]='a'
TypeError: 'str' object does not support item
    assignment
>>> s
'1234567’
```

Ασκήσεις

Φύλλο Εργασίας 2

Άσκηση 2.1

Δημιουργήστε συνάρτηση Python με παράμετρο N που όταν την καλείτε εμφανίζει το όνομά σας τόσες φορές όσες λέει η παράμετρος.

Λύση

```
def print_name (N):
    for i in range(N):
        print('Γιώργος')

print_name(8)
```

Άσκηση 2.2

Δημιουργήστε μια συνάρτηση με τρεις παραμέτρους που επιστρέφει το άθροισμα των παραμέτρων της. Στη συνέχεια δημιουργήστε μια συνάρτηση που θα χρησιμοποιεί την προηγούμενη για να υπολογίζει το μέσο όρο τριών αριθμών.

Λύση

```
def sum3 (a,b,c):
    return a+b+c
def mo3(a,b,c):
    return sum3(a,b,c)/3

print(mo3(22,33,44))
```

Άσκηση 2.3

Γράψτε πρόγραμμα το οποίο διαβάζει από το πληκτρολόγιο μια φράση και στη συνέχεια εμφανίζει τη συμβολοσειρά αντικαθιστώντας όλους τους μη αλφαβητικούς χαρακτήρες με το χαρακτήρα '#'. Χρησιμοποιήστε τη μέθοδο **s.isalpha().**

Υπόδειξη: Η εντολή print(c,end='') γράψει τον χαρακτήρα c και παραμένει στην ίδια γραμμή.

Λύση

```
s=input('Δώστε μια φράση ')
print (s)
for c in s:
    if c.isalpha():
        print(c,end='')
    else:
        print('#',end='‘)
```

Άσκηση 2.4

Δημιουργήστε συνάρτηση η οποία δέχεται ως όρισμα μια συμβολοσειρά και επιστρέφει το πλήθος των αριθμητικών χαρακτήρων (ψηφίων) που περιέχει η συμβολοσειρά (χρησιμοποιήστε τη μέθοδο **s.isdigit()**).

Στη συνέχεια γράψτε πρόγραμμα το οποίο διαβάζει από το πληκτρολόγιο μια συμβολοσειρά (φράση) και χρησιμοποιεί την συνάρτηση για να εμφανίσει το πλήθος των ψηφίων της συμβολοσειράς που διάβασε.

Λύση

```
def count_digits(s):
    n=0
    for c in s:
      if c.isdigit() :
          n+=1
    return n
phrasi=input('Δώστε μια φράση ')
print (phrasi)
print ('Πλήθος ψηφιων = ',count_digits(phrasi))
```

Δομές δεδομένων

- Λίστα (list)
  - Πίνακες
- Πλειάδα (tuple)
- Σύνολο (set)
- Λεξικό (dictionary)

Λίστα

- Η βασική δομή δεδομένων της Python
- Συλλογή στοιχείων (συνήθως του ίδιου τύπου) που το καθένα έχει συγκεκριμένη θέση. Η αρίθμηση των στοιχείων αρχίζει από το 0.

names=[“Γιώργος”, “Άννα”, “Νίκος”]

nums=[8, 3, 8, 77, 2]

Λίστα - παραδείγματα

N=[]	Κενή λίστα
nums=[1, 3, 7, 8, 10] **print(nums[0])** **print(nums[-1])** **print(nums[1:3])**	 1 10 [3, 7]
for x in nums: **print(x)**	Εμφανίζει τα στοιχεία της λίστας nums
print(nums)	Εμφανίζει τα στοιχεία της λίστας nums
len(nums)	Μήκος λίστας nums (5)

Λίστα - παραδείγματα

3 in nums	True
4 in nums	False
L1=[‘a’, ’b’] **L2=[‘c’, ’d’]** **M=L1 + L2**	M=[‘a’, ‘b’, ‘c’, ‘d’]
del nums[2]	Διαγραφή στοιχείου
nums.append(18)	Προσθήκη στοιχείου

Πίνακες – υλοποίηση με λίστα

- Η βασική δομή δεδομένων της Python
- Συλλογή στοιχείων (συνήθως του ίδιου τύπου) που το καθένα έχει συγκεκριμένη θέση. Η αρίθμηση των στοιχείων αρχίζει από το 0.

names=[“Γιώργος”, “Άννα”, “Νίκος”]

nums=[8, 3, 8, 77, 2]

Ασκήσεις

Φύλλο Εργασίας 3

Άσκηση 3.2

Γράψτε πρόγραμμα το οποίο διαχωρίζει τα στοιχεία μιας αριθμητικής λίστας σε δύο νέες λίστες, μία για τα θετικά στοιχεία και μία για τα αρνητικά. Οι αριθμοί θα πρέπει να παραμένουν στην σειρά με την οποία δόθηκαν. Χρησιμοποιήστε ως αρχική αριθμητική λίστα την [2, -3, 5, 1, -7, 16, 4]

Υπόδειξη: Δημιουργήστε αρχικά δύο κενές λίστες.

Λύση

```
num_list=[10,40,-7,5,6,-4,-33,21]
l1=[]
l2=[]
for a in num_list:
    if a>=0:
        l1.append(a)
    else:
        l2.append(a)
print(l1)
print(l2)
```

Άσκηση 3.3

Γράψτε πρόγραμμα το οποίο διαβάζει αριθμούς από το πληκτρολόγιο, μέχρι να δοθεί ο αριθμός 0. Στη συνέχεια με τη χρήση λίστας θα εμφανίζει τους αριθμούς σε αντίστροφη σειρά από αυτή με την οποία τους διάβασε.

Υπόδειξη: χρησιμοποιήστε την εντολή

For n in range(αρχή, τέλος, βήμα)

Λύση

```
l1=[]

a=float(input('Δώστε αριθμό. 0 για τερματισμό: '))
while a!=0:
    l1.append(a)
    a=float(input('Δώστε αριθμό. 0 για τερματισμό: '))

for i in range(len(l1)-1,-1,-1):
   print(l1[i])
```

Άσκηση 3.4

Άσκηση στους πίνακες

Δίνεται η παρακάτω συνάρτηση δημιουργίας πίνακα μεγέθους Ν και με αρχικές τιμές μηδενικά:

```
def new_array(N):
    A=[]
    for i in range(N):
        A.append(0)
    return A
```

Χρησιμοποιήστε τη συνάρτηση για να δημιουργήσετε ένα πίνακα Χ που θα περιέχει 8 στοιχεία.

Α) Εισάγετε από το πληκτρολόγιο στοιχεία στον πίνακα Χ

Β) Βρείτε και εμφανίστε το άθροισμα των στοιχείων του Χ

Γ) Βρείτε και εμφανίστε το μέγιστο στοιχείο του Χ

Άσκηση 3.4

Λύση

```
def new_array(N):
    A=[]
    for i in range(N):
        A.append(0)
    return A

X=new_array(5)
for i in range(5):
    X[i]=float(input('Enter item:'))
```

```
s=0
for i in range(5):
    s=s+X[i]
print(s)

max=X[0]
for i in range(5):
    if X[i]>max:
        max=X[i]
print(max)
```

Άσκηση 3.5

Λύση

```
a=[[1, 2, 3], [4, 5 ,6,], [7, 8, 9]]
print(a)
print(a[0])
print(a[1][1])


s=0
for i in range(3):
    for j in range(3):
        s=s+a[i][j]
print(s)
```

Αρχεία

Αρχεία κειμένου

Άνοιγμα αρχείου:

f=open(“file1.txt”, ”w”)

“w” – write

“r” - read

“r+” - read and write

“a” - append

Κλείσιμο αρχείου:

f.close()

Αρχεία

Γράψιμο σε αρχείο

f.write(“ΚΑΛΗΜΕΡΑ”)

Ανάγνωση από αρχείο:

keim=f.read() διαβάζει όλο το περιεχόμενο στη μεταβλητή keim

m=f.readline() διαβάζει μια γραμμή

for line in f: διαβάζει και εμφανίζει μία –

print(line) μία τις γραμμές κειμένου του αρχείου f

Ασκήσεις

Φύλλο Εργασίας 4

Άσκηση 4.1

Γράψτε πρόγραμμα το οποίο δημιουργεί αρχείο κειμένου file1.txt. Γράψτε τρεις γραμμές κειμένου μέσα στο αρχείο.

Λύση

```
f=open('file1.txt','w')
f.write('Python is the best!!!\n')
f.write('abcdef !@#$%^&*\n')
f.write('1234567890')
f.close()
```

Άσκηση 4.2

Γράψτε πρόγραμμα που αντιγράφει τα περιεχόμενα του αρχείου file1.txt σε ένα άλλο αρχείο κειμένου file2.txt.

Λύση

```
f=open('file1.txt','r')
g=open('file2.txt','w')
for c in f:
    g.write(c)
f.close()
g.close()
```

Άσκηση 4.3

Γράψτε συνάρτηση η οποία δέχεται ως όρισμα το όνομα ενός αρχείου. Η συνάρτηση ανοίγει το αρχείο, διαβάζει τα περιεχόμενα και μετράει και εμφανίζει το πλήθος των ειδικών χαρακτήρων που περιέχει το αρχείο, δηλαδή των χαρακτήρων που δεν είναι γράμματα ή ψηφία (χρησιμοποιήστε τις μεθόδους c.isdigit() και c.isalpha()).

Χρησιμοποιήστε τη συνάρτηση για να μετρήσετε και να εμφανίσετε το πλήθος ειδικών χαρακτήρων που περιέχει το αρχείο file1.txt της άσκησης 4.1.

Άσκηση 4.3

Λύση

```
def count_special(filename):
    f=open(filename,'r')
    n=0
    for line in f:
        for c in line:
            if (not c.isdigit()) and (not c.isalpha()):
                n=n+1
    f.close()
    return(n)

print('Ειδικοί χαρακτήρες = ',count_special('file1.txt'))
```