ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΑΤΡΩΝ
UNIVERSITY OF PATRAS

ΑΝΟΙΚΤΑ ακαδημαϊκά μαθήματα ΠΠ

Εισαγωγή στους Υπολογιστές

Ενότητα 3: Δομημένος Προγραμματισμός, Συναρτήσεις, math, random, δομές επανάληψης

Αβούρης Νικόλαος

Πολυτεχνική Σχολή

Τμήμα Ηλεκτρολόγων Μηχανικών και Τεχνολογίας Υπολογιστών

Σκοποί ενότητας

- Δομημένος Προγραμματισμός, Συναρτήσεις, math, random, δομές επανάληψης

Περιεχόμενα ενότητας

- Δομημένος Προγραμματισμός, Συναρτήσεις, math, random, δομές επανάληψης

Δομημένος Προγραμματισμός, Συναρτήσεις, math, random, δομές επανάληψης

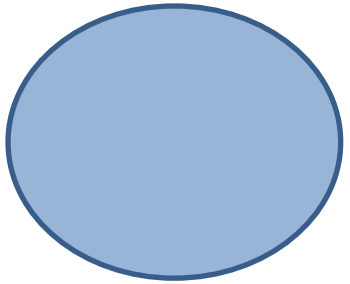
Συναρτήσεις

```
def ονομα (παράμετροι) :
        εντολές
        .....
        return τιμή
```

Παράδειγμα

```
def calcSphereVolume(r):
    volume = (4.0/3.0)* math.pi* \math.pow(r,3)
    return volume

x = calcSphereVolume(r)
```



$V=(4/3)\pi r^3$

Εξηγήστε τις συναρτήσεις

```
def kyklos(radius):
    s = 3.14159265359 * radius**2
    return s
def tetragwno(pleyra):
    return pleyra* pleyra
def trigwno(ypsos, bash):
    s= ypsos* bash * 0.5
    return s
```

Κλήση συνάρτησης

```
rad=4.0

print " Η επιφάνεια κύκλου ακτίνας %.2f m είναι: %.2f
    m2 ενώ \τετραγώνου πλευράς ίδιου μήκους είναι
    %.2f m2" \% (rad, kyklos(rad), tetragwno(rad))
```

Η επιφάνεια κύκλου ακτίνας 4.00 m είναι: 50.27 m2 ενώ τετραγώνου πλευράς ίδιου μήκους είναι 16.00 m25

Τοπικές και καθολικές μεταβλητές

```
a = 42
def f():
    a = 27
    return a
################
b = f()
print a
```

```
a = 42
def f():
    global a
    a = 27
    return a
################
b = f()
print a
```

Τί τυπώνει το πρόγραμμα;

```
def f(x):
    y = x + 5
y = f(4)
print y
```

Απαντήσεις

(α) 9

(β)-1

(γ)None

(δ)Any

Συναρτήσεις-math

```
import math
x = math.sqrt(16)          #square root (x==4)
x = math.pow(2,3)          #equivalent to 23 (x==8)
x = math.ceil(4.3)         #rounds up (x==5)
x = math.floor(7.8)        #rounds down (x==7)
x = math.log(34)           #loge34 (x==3.52)
x = math.log10(100)        #log10100 (x==2)
x = math.sin(3.14)         #angle in radians
x = math.cos(math.pi)
x = math.tan(math.pi * 2)
x = math.asin(-1)          #inverse sine
x = math.sinh(1)           #hyperbolic sine
```

Άσκηση: υπολογισμός υποτείνουσας

```
import math
a = float(raw_input( \"Δώσε το μήκοςAB "))
b = float(raw_input( \"Δώσε το μήκοςAC “))
c = math.sqrt( math.pow(a,2) + \math.pow(b,2) )
print "Μήκος υποτείνουσας:", c
```

Συναρτήσεις: random

```
import random
x = random.randint(0,10)
#δημιουργεί ένα τυχαίο ακέραιομεταξύ
# 0 και10 περιλαμβανομένων
x = random.random()
# δημιουργεί ένα αριθμό κινητής υποδιαστολής >=0.0
    και >=1.0
```

Άσκηση –περιγράψτε τη λειτουργία της συνάρτησης

```
def isInteger(string):
    if string.isdigit():
        return True
    elif string[0] == "-" and \string[1:].isdigit():
        return True
    else:
        return False
```

Αλγοριθμικές δομές επανάληψης

Ερώτηση

- Αν ένας υπολογιστής σήμερα μπορεί να εκτελέσει εκατομμύρια εντολών το δευτερόλεπτο, για να τρέξει ένα πρόγραμμα για ένα δευτερόλεπτο, εγώ πρέπει να γράψω ένα εκατομμύριο εντολές στο πρόγραμμα μου;

Απάντηση

- Η δυνατότητα του υπολογιστή να εκτελέσει πολλές εντολές/ δευτερόλεπτο, στηρίζεται στη δυνατότητα του να εκτελέσει πολλές όμοιες εντολές επαναληπτικά

  –Π.χ. ένας ορθογράφος κειμένου θα ελέγξει μια λέξη και μετά την επόμενη …

  –Ένας video player να ζωγραφίσει ένα pixel ενός frame video και μετά το επόμενο …

Βρόχοι επανάληψης

- Εντολή επανάληψης μιας ομάδας εντολώνμε έλεγχο συνθήκης, ή με απαρίθμηση
- Απαραίτητη η συνθήκη τερματισμού του βρόχου επανάληψης

```
1       x = 0
2       while x < 5:
3               print x
4               x = x + 1
5       print "Done."
```

Ποια η ακολουθία εκτέλεσης των εντολών αυτών και ποιο το αποτέλεσμα;

Βρόχοι επανάληψης

while συνθήκη :

εντολή1

while συνθήκη :

εντολή1

else:

εντολή2

Άσκηση:Πόσες φορές εκτελείται κάθε εντολή του παρακάτω προγράμματος

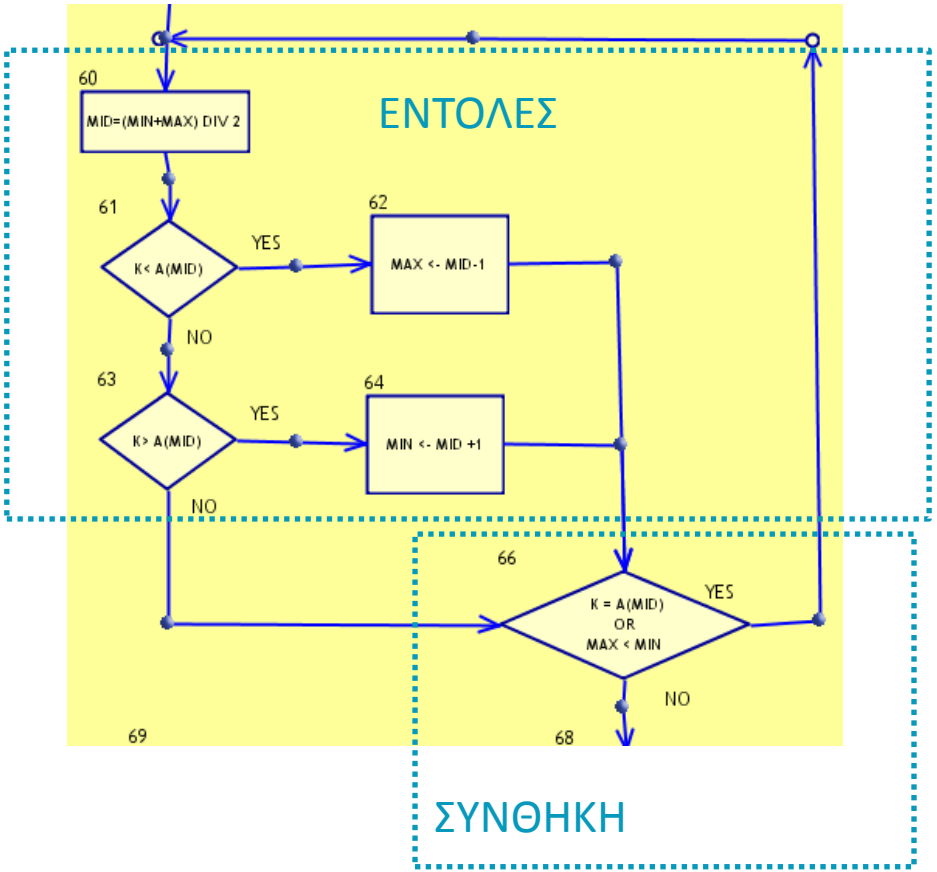
Ποια η λειτουργία του προγράμματος αυτού;

```
n = input()
i = 0
s = 0
while (i <= n) :
    s = s + i
    i = i + 1
print s
```

Άσκηση (Φροντιστήριο)

- Να κατασκευάσετε ένα διάγραμμα ροής και πρόγραμμα python που υπολογίζει με επαναληπτική διαδικασία σε πόσα χρόνια θα γίνετε εκατομμυριούχος αν έχετε ήδη Χ ευρώ και τοκίζετε τα χρήματα σας με επιτόκιο Ε % το χρόνο.

Βρόχος επανάληψης σε διάγραμμα ροής (η συνθήκη στην αρχή ή στο τέλος)



Υπολογισμός s=1+2+…+n

read n

I ←0; s ←0

repeat{

s←s+i

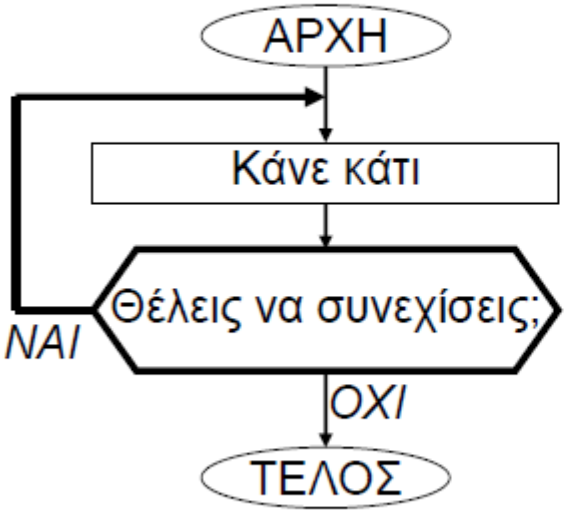
i ←i+1

} until(i>n)

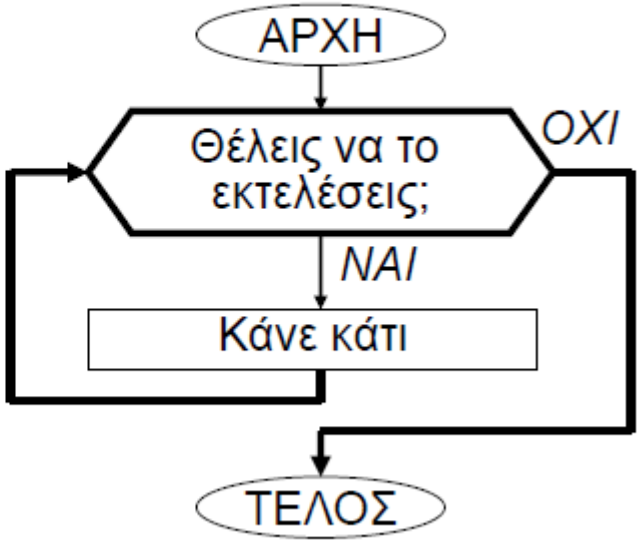
print s

Επαναληπτικές δομές

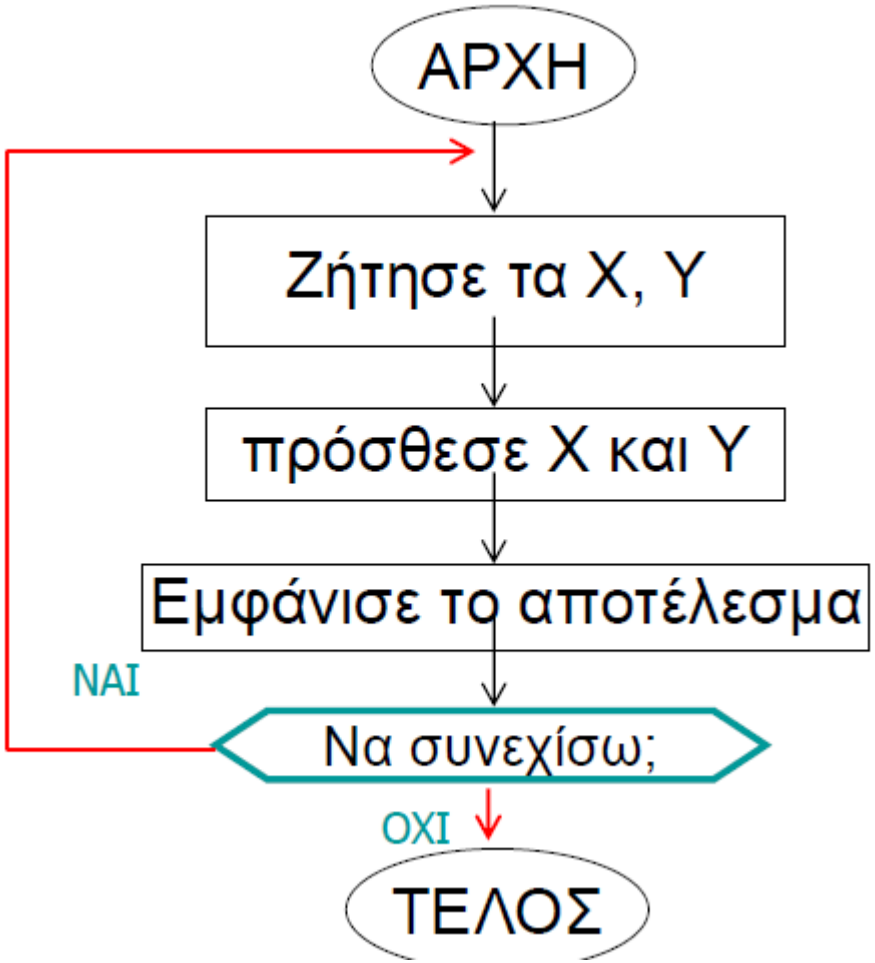
Δομή repeat unitl



Δομή while



Τροποποίηση αθροιστή αριθμών



Αθροιστής αριθμών

Ποια η λειτουργία του προγράμματος αυτού;

```
answer= "y"
    while answer == "y":
        x = float(raw_input("δώσε 1ο αριθμό:"))
        y = float(raw_input("δώσε 2ο αριθμό:" ))
        sum = x + y
        print "Sum:" + str(sum)
        answer= raw_input("να συνεχίσω(y/n)? ")
    print "*** τέλος προγράμματος***"
```

Άσκηση (μέρος Β, Φροντ. 6)

- Γράψτε δύο συναρτήσεις hasLetters(s) και hasDigits(s) οι οποίες δέχονται σαν παράμετρο μια συμβολοσειρά s και η πρώτη γίνεται true αν υπάρχει κάποιος αλφαβητικός χαρακτήρας στην s και η δεύτερη αν υπάρχει αριθμός στην s. Σημείωση : δες τις συναρτήσεις isalpha() και isdigit().

while True: -break
Διακοπή βρόχου όταν συμβεί γεγονός

```
while True:
    userInput = raw_input("Enter an integer:")
    if userInput.isdigit() or userInput[0] == "-" and \
        userInput[1:].isdigit():
        break
    else:
        print "Incorrect input."
```

Βρόχοι επανάληψης με απαρίθμηση (for)

for i in range(start,end,step):

εντολές

Όπου i είναι μεταβλητή απαρίθμησης η οποία παίρνει τιμές από start, σε end-1με βήμα step

Ισοδυναμία δομών for και while

```
for i in range(start,end,step):
        Εντολές
```

```
i = start
while i < end:
        εντολές...
        i = i + step
```

#προσοχή < , όχι <=
(ερώτηση: ισχύει για αρνητικό βήμα;)

Παραδείγματα

```
for i in range(0, 6, 1):
        print i,
#prints 0 1 2 3 4 5
#print the integers from 1 to 15
for i in range(1, 16, 1):
        print i,
```

Παραδείγματα

for i in **range(6):**

print i,

#prints **0 1 2 3 4 5**

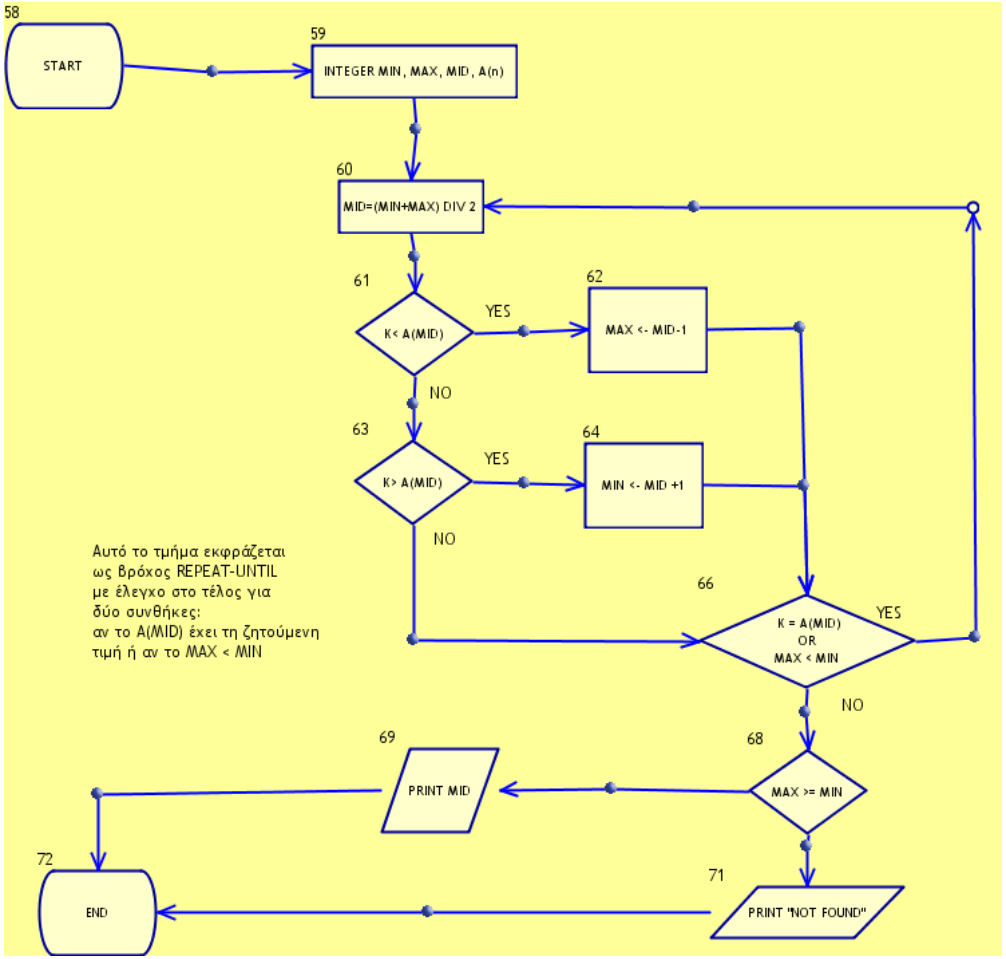
Θεωρεί range (0, 6, 1)

Παραδείγματα

```
# Εκτύπωσε τα τετράγωνα των αριθμών από το 0 μέχρι 5
for i in [0, 1, 2, 3, 4, 5]:
    print "i now equal to:", i
    sumsquares = sumsquares + i**2 # sumsquares incremented here
    print "sum of squares now equal to:", sumsquares
print "------"


# εκτύπωσε τις τιμές από το 0 μέχρι και το 99
for i in range(0, 100):
    print i
```

Αλγόριθμοι: αναπαράσταση διαγράμματος ροής



Συνάρτηση range(a,b,c)

```
>>> range (10)
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
>>> range (5, 10)
[5, 6, 7, 8, 9]
>>> range (0, 10, 3)
[0, 3, 6, 9]
>>> range (-10, -100, -30)
[-10, -40, -70]
```

range(len(a))

```
>>> a = ['Mary', 'had', 'a', 'little', 'lamb']
>>> for i in range(len(a)):
...         printi, a[i]
...
0 Mary
1 had
2 a
3 little
4 lamb
```

else σε βρόχους

```
for i in range (2,n):
        statements1
else:
        statements2
```

Άσκηση

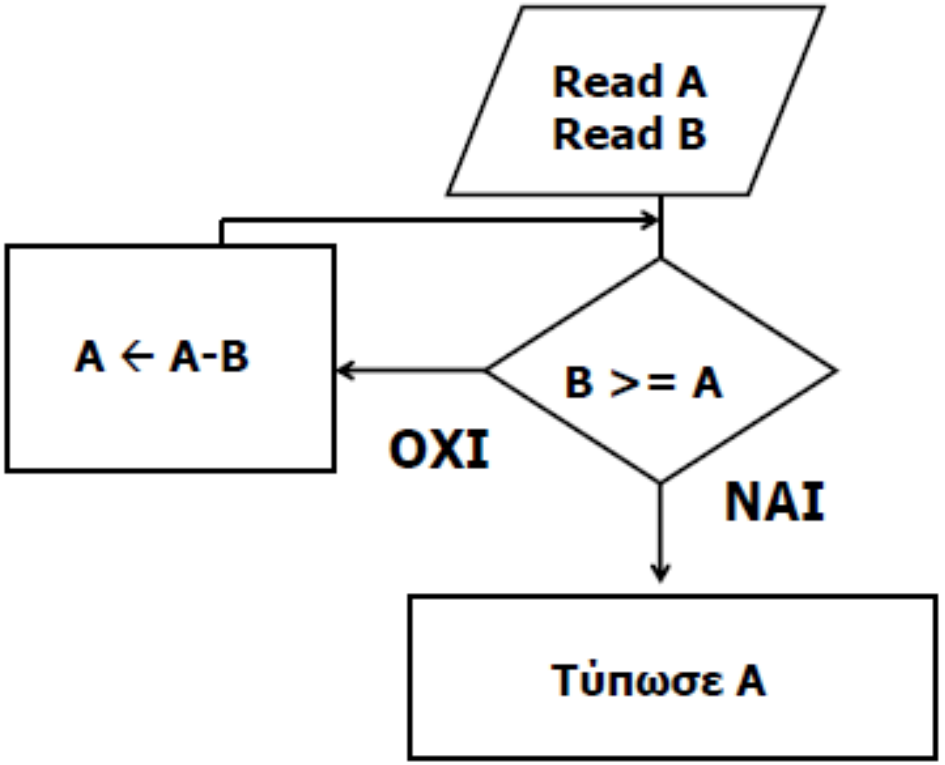
- Να γράψετε σε pythonκαι διάγραμμα ροής τον αλγόριθμο που υπολογίζει την παρακάτω συνάρτησημε χρήση των εντολών while και for :

$$F(n)= \begin{cases} 0\ ,\ n \leq 0 \\ \sum_{i=1}^{n} i*(i+1),\ n > 0 \end{cases}$$

Άσκηση

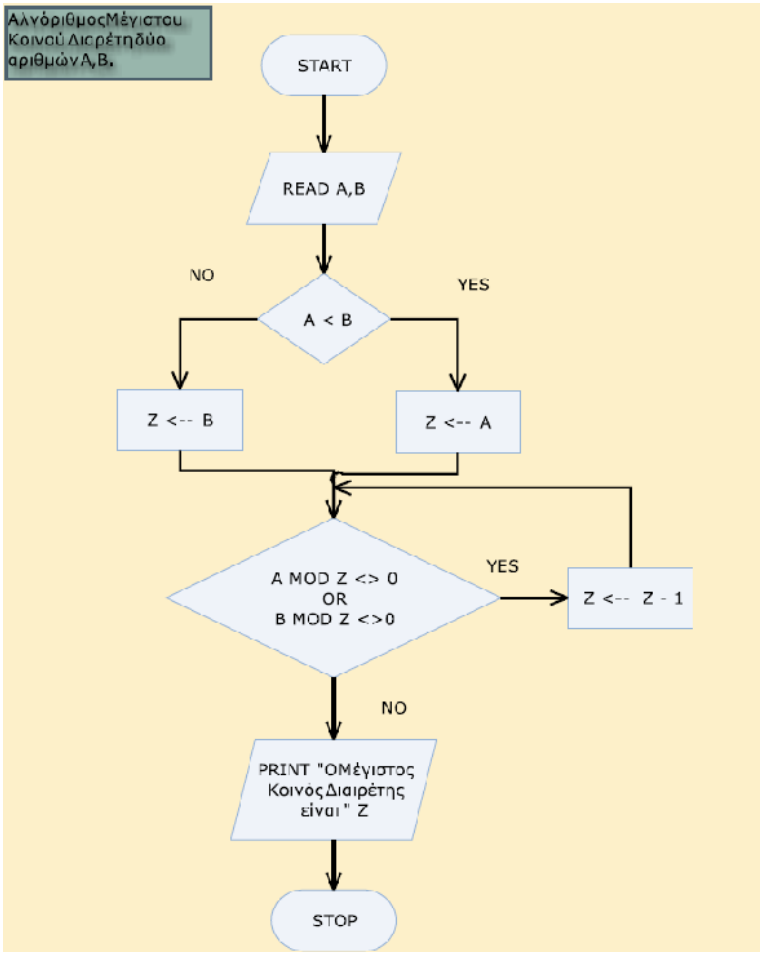
- Να περιγράψετε τον αλγόριθμο που υπολογίζει το **A mod B υπόλοιπο διαίρεσης του Α από το Β.**
- Να περιγράψετε τον αλγόριθμο που υπολογίζει τον **μέγιστο κοινό διαιρέτη δύο αριθμών Α και Β.**
- Να εκφράσετε τους αλγορίθμους αυτούς με python και διάγραμμα ροής

Παράδειγμα: υπόλοιπο A mod B

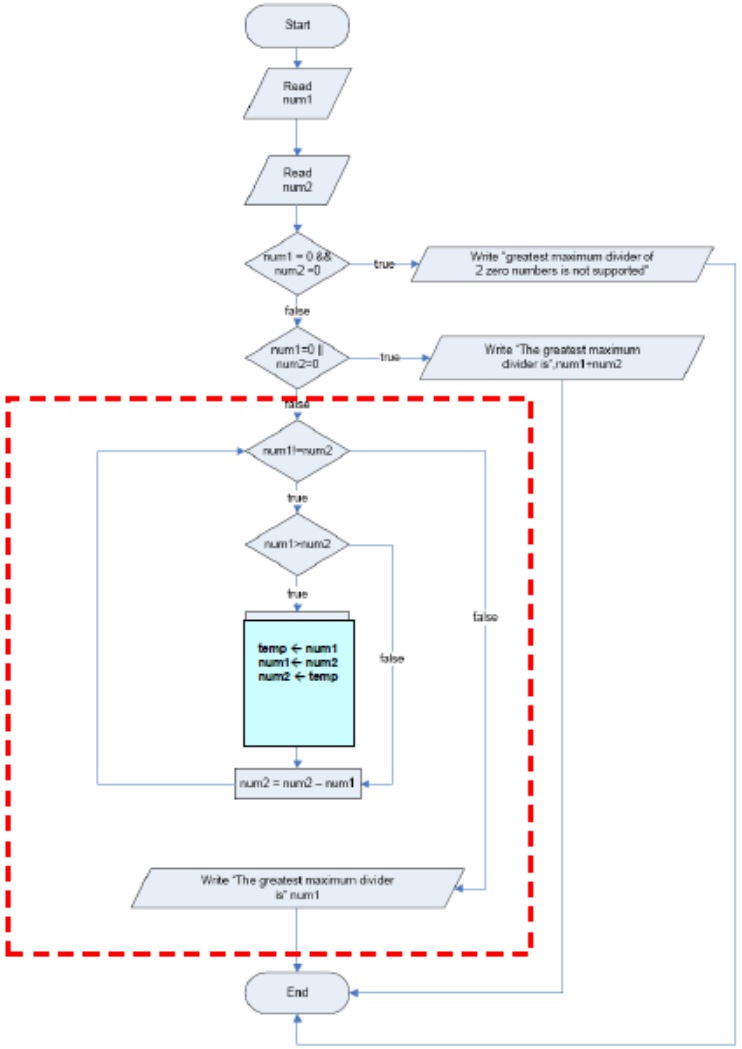


Python: A%B

ΑλγόριθμοςΜέγιστου Κοινού Διαιρέτη(1η έκδοση)



Αλγόριθμος ΜΚΔ (2η έκδοση)



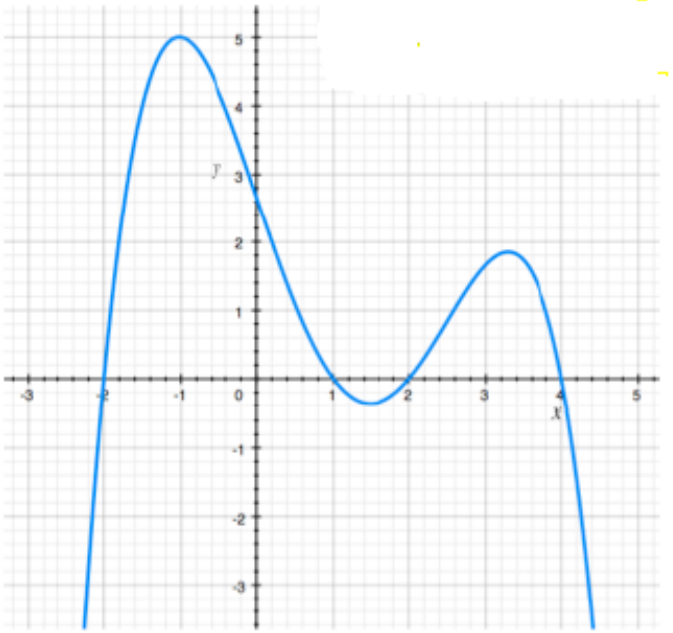
Άσκηση

- Να γράψετε ένα πρόγραμμα Python που βρίσκει τον συνολικό αριθμό εμφάνισης των ψηφίων 0 και 5 σε οποιονδήποτε ακέραιο.
- Πχ Δώσε ένα ακέραιο: 1055030250 αποτέλεσμα : 7

Φροντιστηριακή Άσκηση Φ3: Υπολογισμός Ολοκληρώματος

Στο διπλανό σχήμα φαίνεται η γραφική παράσταση της συνάρτησης:

$$f(x) = -\frac{1}{6}(x-1)*(x-2)*(x+2)*(x-4)$$



Φτιάξτε ένα πρόγραμμα που θα υπολογίζει το ολοκλήρωμα

$$\int_{x_1}^{x_2} f(x)\,dx$$

με τη μέθοδο του τραπεζίου.

Το πρόγραμμα θα ζητά από τον χρήστη να εισάγει 3 παραμέτρους σε μια γραμμή, χωρισμένες με κόμματα:

x1, x2, n

όπου [x1, x2] είναι το διάστημα ολοκλήρωσης και n είναι το πλήθος των τραπεζίων που χρησιμοποιούνται στην προσέγγιση. Το n θα είναι πάντα ακέραιο και μη αρνητικό. Τα x1, x2 θα είναι πραγματικοί αριθμοί με x1<x2.

Το πρόγραμμα θα τυπώνει το ολοκλήρωμα με ακρίβεια 3 δεκαδικών ψηφίων και αμέσως θα διαβάζει νέα τριάδα παραμέτρων επαναλαμβάνοντας τη διαδικασία μέχρι ο χρήστης να δώσει μια τριάδα με n=0.

Ενημερωτικό link για τη μέθοδο του τραπεζίου

http://www.youtube.com/watch?v=8z6JRFvjkpc

Παράδειγμα εισόδου:	Παράδειγμα εξόδου:
-2, 4, 10 -2, 4, 100 -2, 4, 1000 1, 2, 0	10.264 10.795 10.800

Τέλος Ενότητας

Χρηματοδότηση

- Το παρόν εκπαιδευτικό υλικό έχει αναπτυχθεί στο πλαίσιο του εκπαιδευτικού έργου του διδάσκοντα.
- Το έργο «**Ανοικτά Ακαδημαϊκά Μαθήματα στο Πανεπιστήμιο Αθηνών**» έχει χρηματοδοτήσει μόνο την αναδιαμόρφωση του εκπαιδευτικού υλικού.
- Το έργο υλοποιείται στο πλαίσιο του Επιχειρησιακού Προγράμματος «Εκπαίδευση και Δια Βίου Μάθηση» και συγχρηματοδοτείται από την Ευρωπαϊκή Ένωση (Ευρωπαϊκό Κοινωνικό Ταμείο) και από εθνικούς πόρους.

Σημειώματα

Σημείωμα Ιστορικού Εκδόσεων Έργου

Το παρόν έργο αποτελεί την έκδοση 1.0.

Έχουν προηγηθεί οι κάτωθι εκδόσεις:

- Έκδοση 1.0 διαθέσιμη εδώ.

Σημείωμα Αναφοράς

Σημείωμα Αδειοδότησης

Διατήρηση Σημειωμάτων

Οποιαδήποτε αναπαραγωγή ή διασκευή του υλικού θα πρέπει να συμπεριλαμβάνει:

- το Σημείωμα Αναφοράς
- το Σημείωμα Αδειοδότησης
- τη δήλωση Διατήρησης Σημειωμάτων
- το Σημείωμα Χρήσης Έργων Τρίτων (εφόσον υπάρχει)

μαζί με τους συνοδευόμενους υπερσυνδέσμους.

Σημείωμα Χρήσης Έργων Τρίτων

Το Έργο αυτό κάνει χρήση των ακόλουθων έργων:

Εικόνες/Σχήματα/Διαγράμματα/Φωτογραφίες

Διαφάνειες 32: χρήση του Python Editor IDLE

Διαφάνειες 33, 39, 40, 41: χρήση σχεδιαστικού εργαλείου Synergo