**ΕΠΑΛ ΤΗΝΟΥ**

**ΤΟΜΕΑΣ: ΠΛΗΡΟΦΟΡΙΚΗΣ**

**ΕΙΔΙΚΟΤΗΤΑ: ΤΕΧΝΙΚΟΣ Η/Υ & ΔΙΚΤΥΩΝ Η/Υ**

**Για το μάθημα: ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΣ ΥΠΟΛΟΓΙΣΤΩΝ**

# ΑΣΚΗΣΕΙΣ ΣΤΟΝ ΑΝΤΙΚΕΙΜΕΝΟΣΤΡΕΦΗ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟ ΣΕ

# PYTHON

ΚΩΝΣΤΑΝΤΙΝΟΣ ΤΣΙΝΤΖΙΛΙΔΑΣ
ΕΚΠΑΙΔΕΥΤΙΚΟΣ ΚΛ. ΠΕ86
ΠΛΗΡΟΦΟΡΙΚΗΣ

**ΑΣΚΗΣΕΙΣ ΣΤΟΝ ΑΝΤΙΚΕΙΜΕΝΟΣΤΡΕΦΗ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟ ΜΕ ΡΥΤΗΟΝ**

## ΑΣΚΗΣΕΙΣ ΠΡΟΣ ΕΠΙΛΥΣΗ

7.22. Να ορίσετε την κλάση House και τον κατασκευαστή της, που περιγράφει ένα σπίτι με τις εξής ιδιότητες: τον ιδιοκτήτη του σπιτιού, τη διεύθυνση του, τον ταχυδρομικό κώδικα και το εμβαδόν του.

7.23. Να ορίσετε κλάση με όνομα Bouno η οποία έχει:
- α. τρεις ιδιότητες, μία για την ονομασία του βουνού, μια για το υψόμετρο του και μια για την περιοχή που βρίσκεται,
- β. μια μέθοδο αρχικοποίησης των παραπάνω ιδιοτήτων,
- γ. μια μέθοδο εμφάνισης των τιμών των ιδιοτήτων της κλάσης.

7.24. Να ορίσετε κλάση που περιγράφει μια αμαξοστοιχία με όνομα Traino η οποία έχει:
- α. τις ιδιότητες: ονομασία αμαξοστοιχίας, αριθμό βαγονιών, αριθμό επιβατών, ταχύτητα,
- β. μια μέθοδο αρχικοποίησης των παραπάνω ιδιοτήτων, δίνοντας αρχική τιμή μηδέν στην ταχύτητα,
- γ. μια μέθοδο εμφάνισης των τιμών των ιδιοτήτων της κλάσης,
- δ. μια μέθοδο τροποποίησης της ταχύτητας της αμαξοστοιχίας σύμφωνα με την τιμή που δέχεται ως παράμετρο.

**ΑΣΚΗΣΕΙΣ ΣΤΟΝ ΑΝΤΙΚΕΙΜΕΝΟΣΤΡΕΦΗ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟ ΜΕ PYTHON**

**# ΑΣΚΗΣΗ 7.22 OOP**

```
class House:
    def __init__(self,idioktitis,address,tk,emvadon):
        self.idioktitis=idioktitis
        self.address=address
        self.tk=tk
        self.emvadon=emvadon
```

**# ΑΣΚΗΣΗ 7.23 OOP**

```
class Bouno:
    def __init__(self,onomasia,ypsometro,perioxi):
        self.onomasia=onomasia
        self.ypsometro=yprometro
        self.perioxi=perioxi
```

```
    def display():
     print "ONOMASIA:",self.onomasia
     print "YPSOMETRO:",self.ypsometro
     print "PERIOXH:",self.perioxi
```

**# ΑΣΚΗΣΗ 7.24 OOP**

```
class Traino:
  def __init__(self,onomasia,ar_vagoniwn,ar_epivatwn):
      self.onomasia=onomasia
      self.ar_vagoniwn=ar_vagoniwn
      self.ar_epivatwn=ar_epivatwn
      self.taxitita=0
```

```
    def display():
        print "ONOMASIA        :",self.onomasia
        print "ARITHMOS BAGONIWN:",self.ar_vagoniwn
        print "ARITHMOS EPIBATWN:",self.ar_epivatwn
        print "TAXYTHTA        :",self.taxitita

    def change_speed(self,posotita):
        self.taxitita=self.taxitita+posotita
```

**ΑΣΚΗΣΕΙΣ ΣΤΟΝ ΑΝΤΙΚΕΙΜΕΝΟΣΤΡΕΦΗ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟ ΜΕ ΡΥΤΗΟΝ**

7.25. α. Να ορίσετε κλάση με ονομασία Laptop που περιγράφει έναν φορητό ηλεκτρονικό υπολογιστή και έχει:

1. τις ιδιότητες: μοντέλο, μέγεθος οθόνης σε ίντσες, βάρος σε κιλά,
2. μια μέθοδο αρχικοποίησης των παραπάνω ιδιοτήτων,
3. μια μέθοδο εμφάνισης των τιμών των ιδιοτήτων,

β. Να γράψετε τις κατάλληλες εντολές οι οποίες:

1. δημιουργούν ένα αντικείμενο της κλάσης με την ονομασία LP και με ιδιότητες: οθόνη 15,6 ίντσες, βάρος 2,7 κιλά,
2. με τη βοήθεια της κατάλληλης μεθόδου εμφανίζουν τις ιδιότητες του αντικειμένου.

**# ΑΣΚΗΣΗ 7.25 OOP**

```
class Laptop:
    def __init__(self,modelo,scrsize,varos):
        self.modelo=modelo
        self.scrsize=scrsize
        self.varos=varos
    def display(self):
        print "MODELO                  :",self.modelo
        print "MEGETHOS OTHONHS:",self.scrsize,"'''"
        print "BAROS                   :",self.varos,"Kgr"
LP=Laptop("LP",15.6,2.7)
LP.display()
```

**7.26.** Δίνεται η διπλανή κλάση. Να γράψετε τις κατάλληλες εντολές οι οποίες:

α. Δημιουργούν ένα αντικείμενο με όνομα Katastima1, που περιγράφει ένα κατάστημα 120 τ.μ. με δύο ορόφους.

β. Εμφανίζουν τις ιδιότητες του παραπάνω αντικειμένου.

```
class Katastima:
  def __init__(self, metra, orofoi):
    self.metra = metra
    self.orofoi = orofoi
```

**7.27.** Δίνεται η διπλανή κλάση.

α. Να γράψετε ποιος είναι ο κατασκευαστής της κλάσης.

β. Να γράψετε τις ιδιότητες της κλάσης, τις μεθόδους της και τις ενέργειες που πραγματοποιούν.

γ. Να προσθέσετε μια ιδιότητα που αντιπροσωπεύει το φύλο του υπαλλήλου και αρχικοποιείται στον κατασκευαστή.

δ. Να δημιουργήσετε το παρακάτω στιγμιότυπο της κλάσης: αντικείμενο Ypallilos1 με όνομα «Καρακατσάνης», ηλικία 33 και φύλο «Άντρας».

ε. Να καλέσετε την κατάλληλη μέθοδο ώστε να εμφανίζει το όνομα και την ηλικία του υπαλλήλου.

```
class Ypallilos:
  def __init__(self, onoma, hlikia):
    self.onoma = onoma
    self.hlikia = hlikia
    self.idiotita= ' Πωλητής'

  def display (self):
    print 'Ονομα =', self.onoma
    print 'Ηλικία =' self.hlikia

  def setIdiotita (self, idiotita):
    self.idiotita = idiotita
```

**# ΑΣΚΗΣΗ 7.26 OOP**

```
class Katastima:
    def __init__(self,metra,orofoi):
        self.metra=metra
        self.orofoi=orofoi

Katastima1=Katastima(120,2)
print Katastima1.metra,"t.m,"
print Katastima1.orofoi,"orofoi"
```

## #ΑΣΚΗΣΗ 7.27 OOP

Α) κατασκευαστής: Η def __init__(self,onoma,hlikia)

Β) ιδιότητες κλάσης: onoma η οποία περιγράφει το όνομα ενός υπαλλήλου

Hlikia η οποία περιγράφει την ηλικία του υπαλλήλου

Idiotita η οποία περιγράφει την ιδιότητα του υπαλλήλου και έχει αρχική τιμή «Πωλητής»

Γ) η προσθήκη της ιδιότητας «φύλο», αλλάζει την __init ως εξής:

```
def __init(self,onoma,hlikia,fylo):
    Self.onoma=onoma
    self.hlikia=hlikia
    self.fylo=fylo
    self.idiotita=”Πωλητής”
```

Δ)στιγμιότυπο: Ypallilos1=Ypallilos(“Καρακατσάνης”,33,”Άντρας”)

Ε) Ypallilos1.display()

7.28. Δίνεται η διπλανή κλάση.

α. Να γράψετε τις ιδιότητες και τις μεθόδους της κλάσης.

β. Να προσθέσετε μια ιδιότητα που αντιπροσωπεύει το χρώμα του ποδηλάτου και αρχικοποιείται στον κατασκευαστή.

γ. Να τροποποιήσετε τη μέθοδο strofi ώστε να δέχεται ως παράμετρο μια συμβολοσειρά που ορίζει αν το ποδήλατο στρίβει αριστερά ή δεξιά.

δ. Να δημιουργήσετε τα παρακάτω στιγμιότυπα της κλάσης:

1. Αντικείμενο με όνομα PodBounou με τύπο ποδηλάτου «Βουνού», χρώματος «Άσπρο/Γκρι».
2. Αντικείμενο με όνομα PodPolis με τύπο ποδηλάτου «Πόλης», χρώματος «Άσπρο/Μπλε».

ε. Να καλέσετε την κατάλληλη μέθοδο ώστε το αντικείμενο PodBounou να στρίψει αριστερά.

ζ. Να καλέσετε την κατάλληλη μέθοδο ώστε το αντικείμενο PodPolis να τρέχει με 40 χιλ/ώρα.

```
class Podilato:
  def __init__(self, typos):
      self.typos = typos
      self.taxitita = 20

  def strofi (self, taxitita):
      self.taxitita = taxitita
      print 'Η ταχύτητα είναι' self.taxitita

  def strofi(self):
      print 'Το ποδήλατο στρίβει :'
```

## #ΑΣΚΗΣΗ 7.28 OOP

Α) ΙΔΙΟΤΗΤΕΣ ΚΛΑΣΗΣ: typos,taxitita  ΜΕΘΟΔΟΙ ΚΛΑΣΗΣ:speed(self,taxitita) , strofi(self)

Β) προσθήκη ιδιότητας xroma

```
Def __init__(self,typos,xroma):
    Self.typos=typos
    self.xroma=xroma
    self.taxitita=20
```

Γ) τροποποίηση της μεθόδου strofi:

```
Def strofi(self,turn):
    print “Το ποδήλατο στρίβει:”,turn
```

Δ) τα στιγμιότυπα:

```
PodBounou=Podilato(“Βουνού”,”Άσπρο/Γκρι”)
PodPolis=Podilato(“Πόλης”,”Άσπρο/Μπλε”)
```

Ε) PodBounou.strofi(“aristera”)

Ζ) PodPolis.speed(40)

# ΑΣΚΗΣΕΙΣ ΣΤΟΝ ΑΝΤΙΚΕΙΜΕΝΟΣΤΡΕΦΗ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟ ΜΕ ΡΥΤΗΟΝ

**7.29.** Η παρακάτω κλάση διαχειρίζεται μια αποθήκη η οποία χαρακτηρίζεται από τον διαθέσιμο όγκο του αποθηκευτικού της χώρου σε κυβικά μέτρα (κ.μ.). Κάθε φορά που εισάγεται ένα κιβώτιο στην αποθήκη, μειώνεται ο αποθηκευτικό της χώρος, τόσο όσο είναι ο όγκος του κιβωτίου, ενώ αντίστοιχα κάθε φορά που εξάγεται ένα κιβώτιο αυξάνεται ο αποθηκευτικό της χώρος. Να γράψετε τις κατάλληλες εντολές οι οποίες χρησιμοποιώντας τις μεθόδους της κλάσης Apothiki:

α. Δημιουργούν ένα αντικείμενο αποθήκης με όνομα apothiki, με διαθέσιμο αποθηκευτικό χώρο 1500 κ.μ. .

β. Εισάγουν δύο κιβώτια όγκου 200 και 300 κ.μ. .

γ. Εμφανίζουν τον διαθέσιμο αποθηκευτικό χώρο της αποθήκης.

δ. Αφαιρούν ένα κιβώτιο όγκου 200 κ.μ. .

ε. Εμφανίζουν τον διαθέσιμο αποθηκευτικό χώρο της αποθήκης.

```
class Apothiki:
  def __init__(self, ogkos):
    self.ogkos = ogkos
  def eisagogi (self, kivotio):
    self.ogkos -= kivotio
  def ejagogi (self, kivotio):
    self.ogkos += kivotio
  def emfanise (self):
    print 'Χώρος =' self.ogkos
```

**7.30.** α. Να ορίσετε κλάση με το όνομα Arithmos η οποία θα έχει την ιδιότητα timi και τις παρακάτω μεθόδους:

- Eisagogi_timis(timi) η όποια αναθέτει τιμή στην ιδιότητα timi,
- Emfanise_timi() η οποία εμφανίζει την τιμή της ιδιότητας.

β. Να δημιουργήσετε στιγμιότυπο της κλάσης Arithmos με το όνομα Perittos και να χρησιμοποιήσετε τις παραπάνω μεθόδους για να δώσετε την τιμή 11 στην ιδιότητα του αντικειμένου και να την εμφανίσετε.

**ΑΣΚΗΣΕΙΣ ΣΤΟΝ ΑΝΤΙΚΕΙΜΕΝΟΣΤΡΕΦΗ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟ ΜΕ ΡΥΤΗΟΝ**

**#ΑΣΚΗΣΗ 7.29 OOP**

Α) ΔΗΜΙΟΥΡΓΊΑ ΑΝΤΙΚΕΙΜΕΝΟΥ: apothiki=Apothiki(1500)

Β) ΕΙΣΑΓΩΓΗ ΑΝΤΙΚΕΙΜΕΝΩΝ:

apothiki.eisagogi(200)

apothiki.eisagosi(300)

Γ) ΕΜΦΑΝΙΣΗ ΔΙΑΘΕΣΙΜΟΥ ΑΠΟΘΗΚΕΥΤΙΚΟΥ ΧΩΡΟΥ : apothiki.emfanise()

Δ) ΑΦΑΙΡΕΣΗ ΚΙΒΩΤΙΟΥ : apothiki.ejagogi(200)

Ε) ΕΜΦΑΝΙΣΗ ΔΙΑΘΕΣΙΜΟΥ ΑΠΟΘΗΚΕΥΤΙΚΟΥ ΧΩΡΟΥ: apothiki.emfanise()

**ΑΣΚΗΣΕΙΣ ΣΤΟΝ ΑΝΤΙΚΕΙΜΕΝΟΣΤΡΕΦΗ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟ ΜΕ ΡΥΤΗΟΝ**

**#ΑΣΚΗΣΗ 7.30 OOP**

Α)

```
Class Arithmos:
    Def  __init__(self):
        Self.timi=0
    Def Eisagigi_timis(self,timi):
        Self.timi=timi
    Def emfanise_timi(self):
        print “Η τιμή είναι:”,self.timi
```

Β)ΔΗΜΙΟΥΡΓΙΑ ΣΤΙΓΜΙΟΤΥΠΟΥ: Perittos=Arithmos()

ΑΠΟΔΟΣΗ ΤΗΣ ΤΙΜΗΣ 11: Perittos.Eisagogi_timis(11)

ΕΜΦΑΝΙΣΗ ΤΙΜΗΣ: Perittos.emfanise_timi()

7.31. Να γραφεί πρόγραμμα το οποίο:

α. Ορίζει την κλάση Basket η οποία για ένα αγώνα καλαθοσφαίρισης περιέχει τα παρακάτω δεδομένα:
- τα ονόματα των δύο ομάδων που αγωνίζονται,
- τους πόντους που σημείωσε η κάθε μία ομάδα,
- τον νικητή του αγώνα.

και τις παρακάτω μεθόδους:
- init, η οποία κατασκευάζει αντικείμενα της κλάσης και τα αρχικοποιεί,
- ypologismos, η οποία δέχεται τα ονόματα και τους πόντους που σημείωσαν και καταχωρίζει τον νικητή του αγώνα,
- apotelesma, η οποία εμφανίζει τα ονόματα των ομάδων, τους πόντους και τον νικητή του αγώνα.

β. Διαβάζει τα ονόματα των δύο ομάδων και τους πόντους που σημείωσαν σε έναν αγώνα.

γ. Δημιουργεί το αντικείμενο agonas με τα δεδομένα του ερωτήματος β.

δ. Εμφανίζει τον νικητή του αγώνα χρησιμοποιώντας τις κατάλληλες μεθόδους.

7.32. Να γραφεί πρόγραμμα το οποίο:

α. Ορίζει την κλάση Ergostasio, η οποία διαχειρίζεται ένα εργοστάσιο παραγωγής εμφιαλωμένου νερού και περιλαμβάνει τα παρακάτω δεδομένα:
- τον κωδικό αριθμό της εμφιάλωσης,
- τον αριθμό μπουκαλιών που εμφιαλώθηκαν σε διάστημα 7 ημερών με τη μορφή μιας λίστας,

και τις παρακάτω μεθόδους:

- Μία μέθοδο η οποία κατασκευάζει αντικείμενα της κλάσης και τα αρχικοποιεί.
- Μία μέθοδο, η οποία εμφανίζει τα παραπάνω δεδομένα.
- Μία μέθοδο η οποία υπολογίζει και επιστρέφει τη μέση ημερήσια παραγωγή των 7 ημερών.

β. Δημιουργεί το αντικείμενο partida με κωδικό αριθμό 999 και τους αριθμούς μπουκαλιών εμφιάλωσης 20, 28, 25, 24, 25,28 29, εκφρασμένους σε χιλιάδες, για κάθε μια ημέρα της εβδομάδας.

γ. Με τη χρήση κατάλληλων μεθόδων της κλάσης:

1. εμφανίζει τον κωδικό αριθμό εμφιάλωσης και τον αριθμό μπουκαλιών για κάθε μια ημέρα της εβδομάδας,
2. υπολογίζει και εμφανίζει τη μέση ημερήσια παραγωγή.

**#ΑΣΚΗΣΗ 7.31 OOP**

```
class Basket:
    def __init__(self,team1,points1,team2,points2):
        self.team1=team1
        self.points1=points1
        self.team2=team2
        self.points2=points2
        self.winner=""
    def ypologismos(self,team1,points1,team2,points2):
        if self.points1>self.points2:
            self.winner=self.team1
        elif self.points1<self.points2:
            self.winner=self.team2
        else:
            self.winner="ISOPALIA"
    def apotelesma(self):
        print"OMADA 1:",self.team1,"PONTOI:",self.points1
        print"OMADA 2:",self.team2,"PONTOI:",self.points2
        print"NIKHTHS AGWNA:",self.winner
```

```
omada1=raw_input("OMADA 1:")
pontoi1=input("PONTOI:")
omada2=raw_input("OMADA 2:")
pontoi2=input("PONTOI:")
agonas=Basket(omada1,pontoi1,omada2,pontoi2)
agonas.ypologismos(omada1,pontoi1,omada2,pontoi2)
agonas.apotelesma()
```

**#ΑΣΚΗΣΗ 7.32**

```
class Ergostasio:
    def __init__(self,code,arbouk):
        self.code=code
        self.arbouk=arbouk[:7]  #ιδιότητα με τιμή λίστα 7 στοιχείων,
                                #αν παραληφθεί το [:7], τότε κάθε λίστα μπορεί να έχει οποιοδήποτε μήκος
```

```
    def display(self):
        print"CODE:",self.code
        print"MPOYKALIA:",self.arbouk

    def ypologismos(self):
        total=0.0
        for item in self.arbouk:
            total+=item
        mo=total/len(self.arbouk)
        return mo
partida0=Ergostasio(999,[20,28,25,24,25,28,29])
partida0.display()
print partida0.ypologismos()
partida1=Ergostasio(888,[20,28,12,24,3,28,6])
partida1.display()
print partida1.ypologismos()
```

7.33. Ένα ρομπότ με δύο πόδια, κινείται με σταθερή ταχύτητα και καλύπτει απόσταση που υπολογίζεται από το γινόμενο του αριθμού βημάτων που πραγματοποιεί επί τον μήκος του ανοίγματος των δύο ποδιών (βήμα) σε εκατοστά. Επίσης καταναλώνει ενέργεια ίση με 10 Joule για κάθε μέτρο που κινείται. Να γραφεί πρόγραμμα το οποίο:

α. Ορίζει την κλάση Robot, η οποία περιλαμβάνει, τα παρακάτω δεδομένα:
- τον αριθμό βημάτων που περπάτησε,
- το μήκος ενός βήματος σε εκατοστά,
- την απόσταση που διήνυσε σε μέτρα,
- την κατανάλωση ενέργειας.

και τις παρακάτω μεθόδους:
- Μία μέθοδο που κατασκευάζει αντικείμενα της κλάσης και αρχικοποιεί τα δεδομένα ως εξής: την απόσταση και την κατανάλωση ενέργειας με μηδενική τιμή, τον αριθμό βημάτων και το μήκος βήματος, σύμφωνα με αυτά που δέχεται ως παράμετρο.
- Μία μέθοδο που εμφανίζει τα παραπάνω δεδομένα.
- Μία μέθοδο που υπολογίζει την απόσταση σε μέτρα και ενημερώνει το αντίστοιχο δεδομένο.
- Μία μέθοδο που υπολογίζει την κατανάλωση ενέργειας και ενημερώνει το αντίστοιχο δεδομένο.

β. Για δύο διαφορετικά ρομπότ, διαβάζει τον αριθμό των βημάτων που πραγματοποίησε το κάθε ρομπότ καθώς και το μήκος του ενός βήματος του σε εκατοστά.

γ. Δημιουργεί τα αντικείμενα Robo100 και Robo101 χρησιμοποιώντας τα δεδομένα του ερωτήματος β.

δ. Με τη χρήση κατάλληλων μεθόδων της κλάσης:
1. υπολογίζει την απόσταση που διήνυσε το κάθε ρομπότ σε μέτρα,
2. υπολογίζει την κατανάλωση ενέργειας του κάθε ρομπότ,
3. εμφανίζει τον αριθμό βημάτων, το μήκος του βήματος, την απόσταση σε μέτρα και την κατανάλωση ενέργεια του κάθε ρομπότ.

ε. Βρίσκει και εμφανίζει το ρομπότ που κινήθηκε σε μικρότερη απόσταση.

```
class Robot:
    def __init__(self,vimata,mikosvimatos):
        self.vimata=vimata
        self.mikosvimatos=mikosvimatos
        self.apostasi=0
        self.katanalosi=0

    def display(self):
        print "ARITHMOS BHMATWN:",self.vimata
        print "MHKOS BHMATOS SE cm:",self.mikosvimatos
        print "APOSTASH POY DIENHSE SE m:",self.apostasi
        print "KATANALWSH ENERGEIAS SE joule/m:",self.katanalosi
        print

    def ypologismos_apostasis_se_m(self,v,mv):
        self.apostasi=v*mv/100.0


    def ypologismos_katanalosis(self):
        self.katanalosi=self.apostasi*10

print "STOIXEIA GIA TO 1o ROBOT"
```

```
print"---------------------------------------"
v1=input("ARITHMOS BHMATWN 1ou ROBOT:")
m1=input("MHKOS BHMATOS SE cm:")
print
print
Robo100=Robot(v1,m1)
print "STOIXEIA GIA TO 2o ROBOT"
print"---------------------------------------"
v2=input("ARITHMOS BHMATWN 2ou ROBOT:")
m2=input("MHKOS BHMATOS SE cm:")
print
print
Robo101=Robot(v2,m2)
Robo100.ypologismos_apostasis_se_m(v1,m1)
```

```
Robo101.ypologismos_apostasis_se_m(v2,m2)
Robo100.ypologismos_katanalosis()
Robo101.ypologismos_katanalosis()
print"APOTELESMATA GIA TO 1o ROBOT"
print"-----------------------------------------"
Robo100.display()
print"APOTELESMATA GIA TO 2o ROBOT"
print"-----------------------------------------"
Robo101.display()
if Robo100.apostasi<Robo101.apostasi:
    print "Se mikroteri apostash kinithike to robot: Robo100"
elif Robo100.apostasi>Robo101.apostasi:
   print "Se mikroteri apostash kinithike to robot: Robo101"
else: print "Kinithikan sthn idia apostash"
```