Python – Μάθημα 6

Turtle (χελώνα)

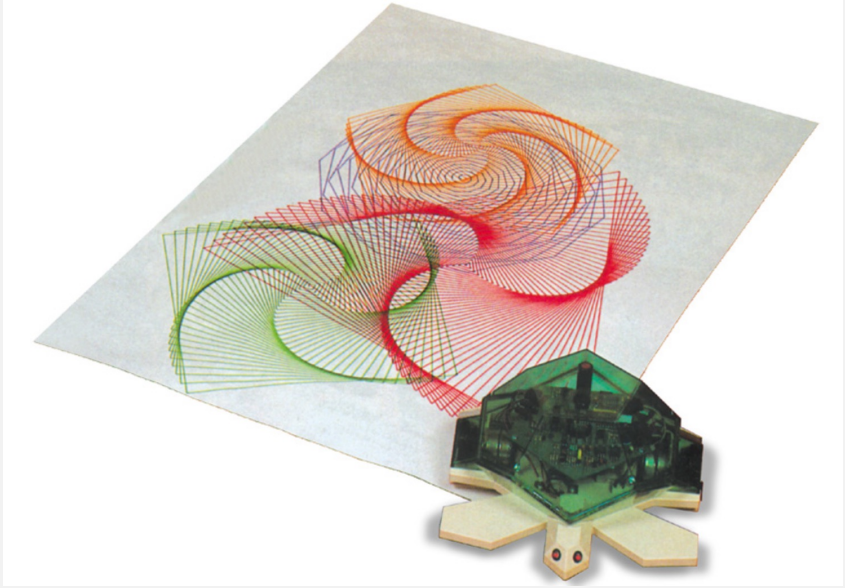
Modules στην Python

Οι δυνατότητες της Python είναι πολλές. Αυτό οφείλεται (και) στο ότι μπορούμε να χρησιμοποιήσουμε έτοιμα **modules** για να δώσουμε πολλές και νέες δυνατότητες στα προγράμματα που δημιουργούμε, χωρίς να χρειαστεί να γράψουμε πολύπλοκο κώδικα.

Module Turtle

Ένα χρήσιμο module είναι το **turtle** (χελώνα). Βασίζεται στο περιβάλλον προγραμματισμού **LOGO** που δημιουργήθηκε το 1967 για να διδάξει μαθηματικά σε παιδιά στις ΗΠΑ (με τη χρήση μιας χελώνας ρομπότ!).

Ρομπότ για προγραμματισμό σε logo



Python όπως λέμε Logo

Η λογική της LOGO (και της Python με τις πρόσθετες εντολές)- είναι απλή: δίνουμε εντολές (FORWARD, LEFT, RIGHT, κτλ) σε μια “χελώνα” στην οθόνη μας. Η “χελώνα” κινείται πάνω στην οθόνη και δημιουργεί σχέδια.

Για να μπορέσουμε να χρησιμοποιήσουμε το module turtle το πρώτο που πρέπει να κάνουμε, είναι να το εισαγάγουμε στο πρόγραμμά μας. Για να το κάνουμε αυτό αρκεί να πληκτρολογήσουμε

```
from turtle import *
```

Βασικές εντολές χελώνας

Οι βασικές εντολές ελέγχου της χελώνας στην Python είναι:

Εντολή	
forward() | **fd**()	Μετακινεί τη χελώνα προς τα **μπροστά** τόσα **pixel** όσο ο αριθμός στην παρένθεση
b**ackward**() | **bk**() | **back**()	Μετακινεί τη χελώνα προς τα **πίσω** τόσα **pixel** όσο ο αριθμός στην παρένθεση
right() | **rt**()	Στρίβει **επιτόπου** τη χελώνα προς τα **δεξιά** τόσες **μοίρες** όσες ο αριθμός στην παρένθεση
left() | **lt**()	Στρίβει **επιτόπου** τη χελώνα προς τα **αριστερά** τόσες **μοίρες** όσες ο αριθμός στην παρένθεση
pendown() | **pd**() | **down**()	Κατεβάζει το στυλό έτσι ώστε κατά την κίνησή της η χελώνα να αφήνει ίχνη
penup() | **pu**() | **up**()	Ανεβάζει το στυλό έτσι ώστε κατά την κίνησή της η χελώνα να μην αφήνει ίχνη
clearscreen()	Καθαρίζει την εικόνα για να ξεκινήσουμε από την αρχή
undo()	Ακυρώνει την εκτέλεση της τελευταίας εντολής

Παράδειγμα

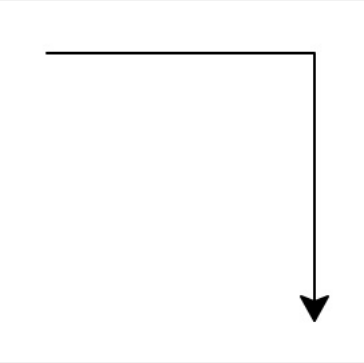
Ας δούμε για παράδειγμα τον παρακάτω απλό κώδικα.

1. Εισάγουμε το module turtle στον κώδικα μας
2. Η χελώνα κινείται προς τα μπροστά κατά 100 pixel. Αρχικά η χελώνα έχει κατεύθυνση προς τα δεξιά και με το στυλό κάτω όποτε μετά την εντολή αυτή σχηματίζεται ένα ευθύγραμμα τμήμα μήκους 100 pixel
3. Η χελώνα κάνει μια επιτόπου δεξιά στροφή 90 μοιρών οπότε πλέον έχει κατεύθυνση προς τα κάτω
4. Η χελώνα κινείται προς τα μπροστά (προς την κατεύθυνση που κοιτάει δηλαδή, σε αυτή την περίπτωση προς τα κάτω) κατά 100 pixel.

Κώδικας

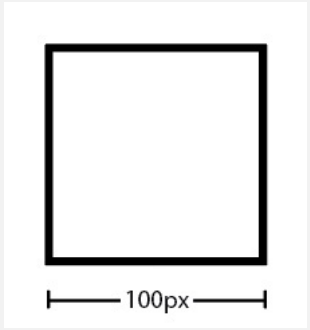
```
1  from turtle import *
2  forward(100)
3  right(90)
4  forward(100)
```

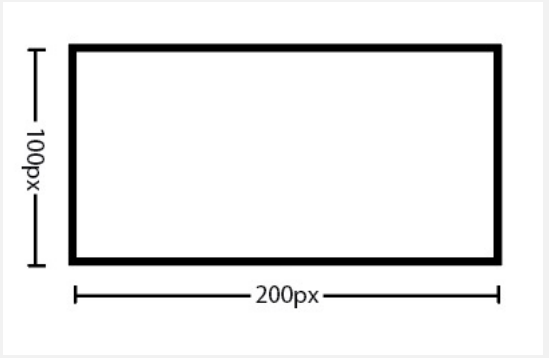
Αποτέλεσμα εκτέλεσης

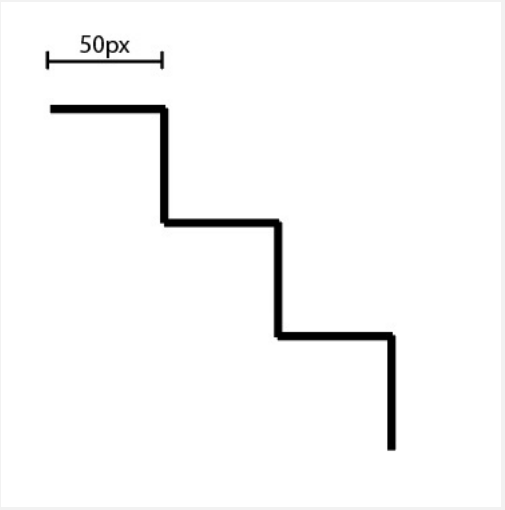


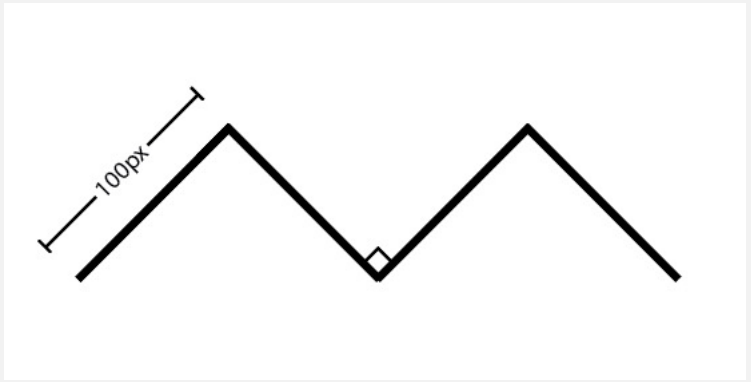
Δραστηριότητα 1

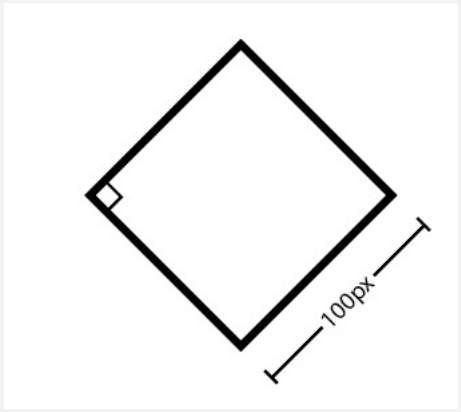
Χρησιμοποιώντας τις εντολές της **python** από το module **turtle** δημιουργήστε τα παρακάτω σχήματα:

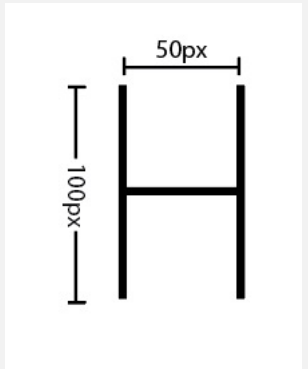


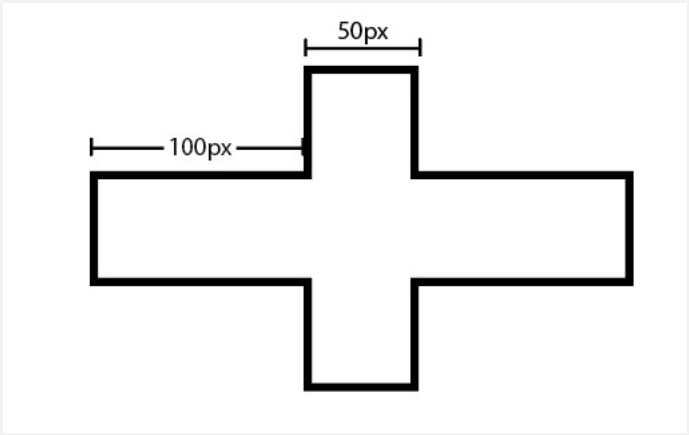






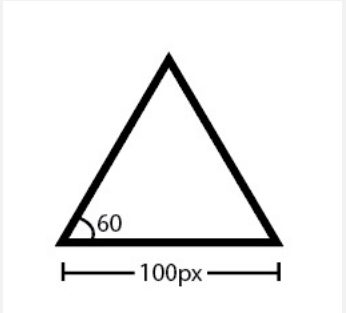






Χελώνα και γωνίες

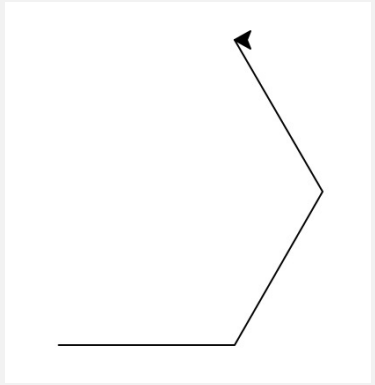
Έστω ότι θέλουμε να δημιουργήσουμε με τη χελώνα ένα **ισόπλευρο τρίγωνο** με πλευρά 100 pixel. Όπως γνωρίζουμε το ισόπλευρο τρίγωνο έχει όλες τις όλες τις γωνίες του ίσες με **60º**. Επομένως δοκιμάζουμε τον παρακάτω κώδικα:



Κώδικας

```
1  from turtle import *
2  forward(100)
3  left(60)
4  forward(100)
5  left(60)
6  forward(100)
7  left(60)
```

Αποτέλεσμα εκτέλεσης

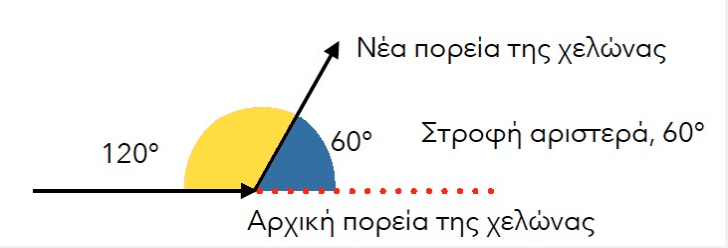


Το αποτέλεσμα προφανώς δεν είναι αυτό που περιμέναμε. Ποιο όμως είναι το λάθος που κάναμε;

Χελώνα και γωνίες

Η γωνία μας δεν είναι η σωστή. Ο λόγος είναι απλός: Η χελώνα μας, “κοίταζε” δεξιά όταν της είπαμε να στρίψει 60° αριστερά. Στην πραγματικότητα, δημιουργήθηκαν δύο γωνίες που είναι **παραπληρωματικές** (το άθροισμά τους είναι 180ο). Η μία είναι η εξωτερική και η άλλη η εσωτερική.

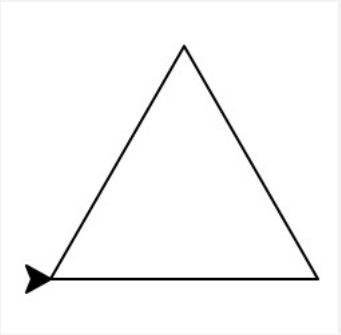
Η χελώνα κάνει πάντα την **εξωτερική** γωνία. Για να δημιουργηθεί λοιπόν **εσωτερική** γωνία 60ο η χελώνα θα πρέπει να στρίψει 120ο.



Επομένως ο κώδικας για το ισόπλευρο τρίγωνο γίνεται:

```
1  from turtle import *
2  forward(100)
3  left(120)
4  forward(100)
5  left(120)
6  forward(100)
7  left(120)
```

Κώδικας



Αποτέλεσμα εκτέλεσης

Χρήση for για δημιουργία σχημάτων

Αν δούμε τον κώδικα για το ισόπλευρο τρίγωνο, εύκολα θα παρατηρήσουμε ότι οι εντολές **forward(100)** και **left(120)** **επαναλβάνονται 3 φορές**. Επομένως μπορούμε να κάνουμε χρήση της for. Σε αυτή την περίπτωση ο κώδικας για το τρίγωνο γίνεται:

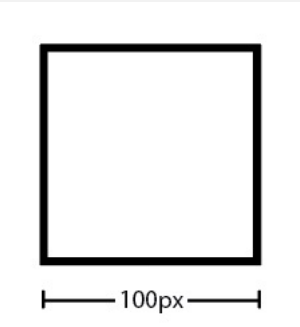
```
1  from turtle import *
2  for x in range(3):
3      forward(100)
4      left(120)
```

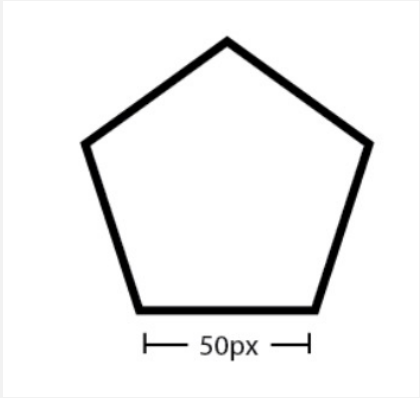
Ένας εύκολος τρόπος να υπολογίζουμε τη γωνία που πρέπει να στρίψει η χελώνα προκειμένου να δημιουργηθεί ένα **κλειστό συμμετρικό σχήμα** είναι ότι ο **αριθμός των επαναλήψεων** * τη **γωνία** πρέπει να μας κάνει **360**.

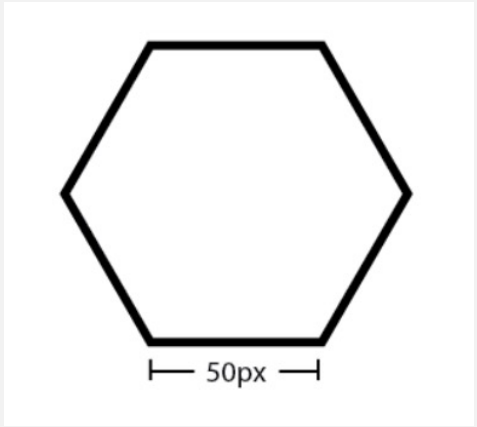
Για παράδειγμα για ένα κανονικό οκτάγωνο η γωνία είναι 360/8=45^{o} .

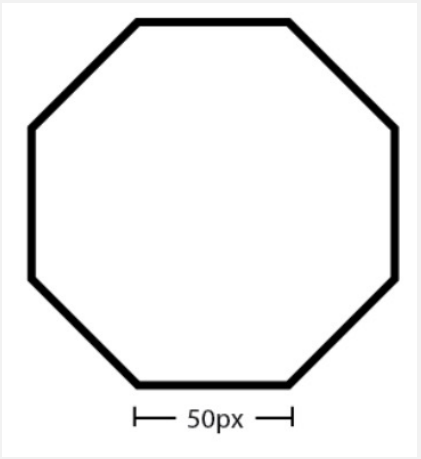
Δραστηριότητα 2

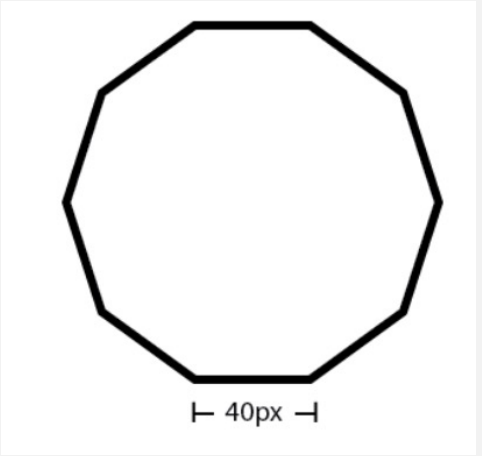
Χρησιμοποιώντας την for δημιουργήστε τα παρακάτω σχήματα:

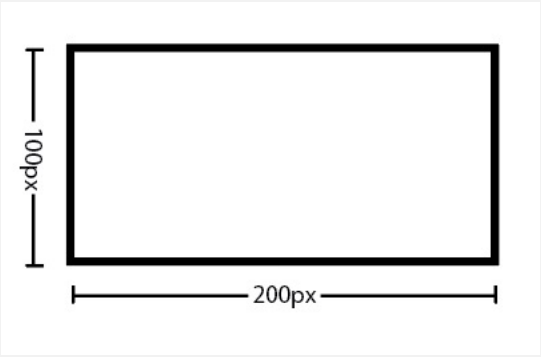


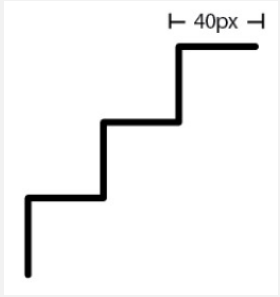












Χρώματα

Αν θέλουμε να αλλάξουμε το χρώμα με το οποίο σχεδιάζει η χελώνα χρησιμοποιούμε την εντολή **color().**

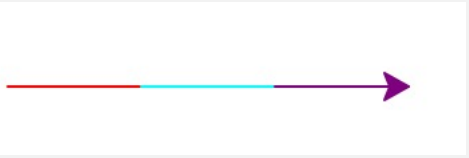
color(όνομα_χρώματος)

Το χρώμα μπορούμε να το επιλέξουμε με την ονομασία του μέσα σε εισαγωγικά. Πχ **black**, **blue**, **red**, **green**, **magenta**, **brown**, **gold**, **teal**, **maroon**, **purple**, **violet**, **pink, navy, indigo, coral, sienna, lime, olive**.

Κώδικας

```
1 from turtle import *
2 color("red")
3 forward(50)
4 color("cyan")
5 forward(50)
6 color("purple")
7 forward(50)
```

Αποτέλεσμα εκτέλεσης



Κύκλος

Για να δημιουργήσουμε έναν κύκλο αρκεί να χρησιμοποιήσουμε την εντολή **circle().**

circle(radius) ή circle(radius, extent)

Η παράμετρος **radius** καθορίζει την ακτίνα του κύκλου σε pixel. Αν η τιμή είναι θετική ο κύκλος σχεδιάζεται αριστερόστροφα (αντίθετα με τους δείκτες του ρολογιού) με ακτίνα radius ενώ αν είναι αρνητική δεξιόστροφα με την ίδια ακτίνα.

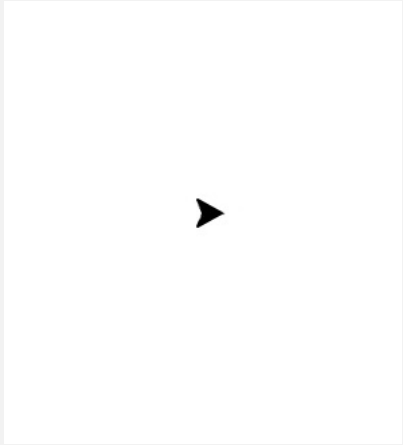
Η παράμετρος **extent**, δεν είναι απαραίτητη. Χρησιμοποιείται μόνο αν θέλουμε να σχεδιάσουμε τμήμα κύκλου. Σε αυτή την περίπτωση γράφουμε τις μοίρες του κύκλου που θέλουμε να σχεδιάσουμε (πχ για ημικύκλιο γράφουμε 180)

Κύκλος (παραδείγματα)

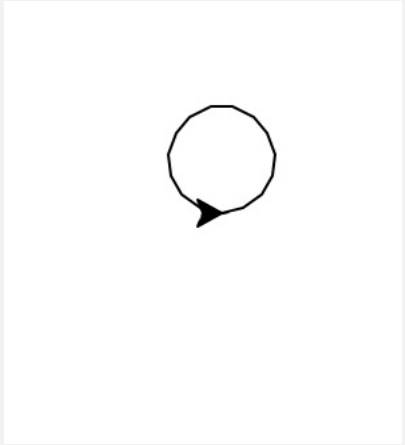
Κώδικας

```
1  from turtle import *
2  circle(20)
3  circle (-40)
```

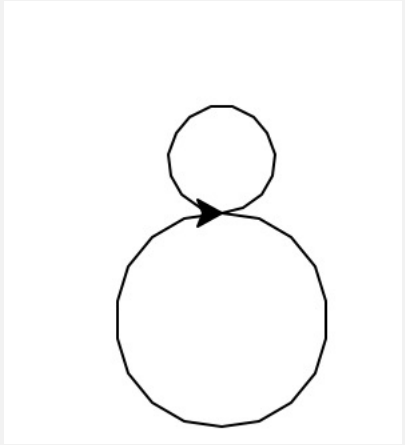
Αρχική θέση χελώνας



Αριστερόστροφα κύκλος ακτίνας 20 px



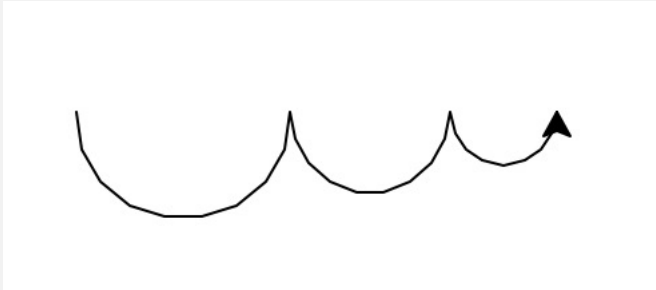
Δεξιόστροφα κύκλος ακτίνας 40 px



Κώδικας

```
1  from turtle import *
2  right(90)
3  circle(40,180)
4  left(180)
5  circle(30,180)
6  left(180)
7  circle(20,180)
```

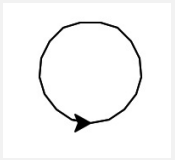
3 αριστερόστροφα ημικύκλια ακτίνας 40, 30 και 20 px.



Πολλοί κύκλοι

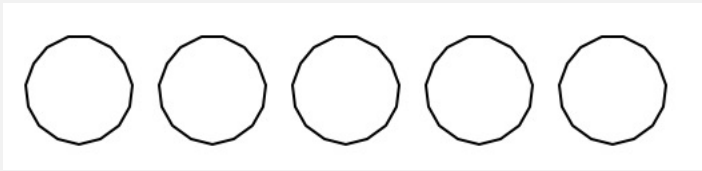
Αν θέλουμε να δημιουργήσουμε **5 κύκλους** τον έναν δίπλα στον άλλο, μπορούμε να χρησιμοποιήσουμε τη **for**. Έτσι για παράδειγμα μια πρώτη προσπάθεια θα μπορούσε να είναι και ο παρακάτω κώδικας:

```
1  from turtle import *
2  for x in range (5):
3      circle(20)
```



Το αποτέλεσμα δεν είναι το επιθυμητό. Παρόλο που η for εκτελείται **5 φορές**, βλέπουμε μόνο έναν κύκλο. Στην ουσία δημιουργούνται 5 κύκλοι αλλά ο ένας πάνω στον άλλο. Για να διορθωθεί αυτό αρκεί να υπάρχει μια μετατόπιση μετά τη δημιουργία ενός κύκλου. Π.χ. θα μετακινούμε το χελωνάκι κατά 50 pixel. Ο κώδικας γίνεται:

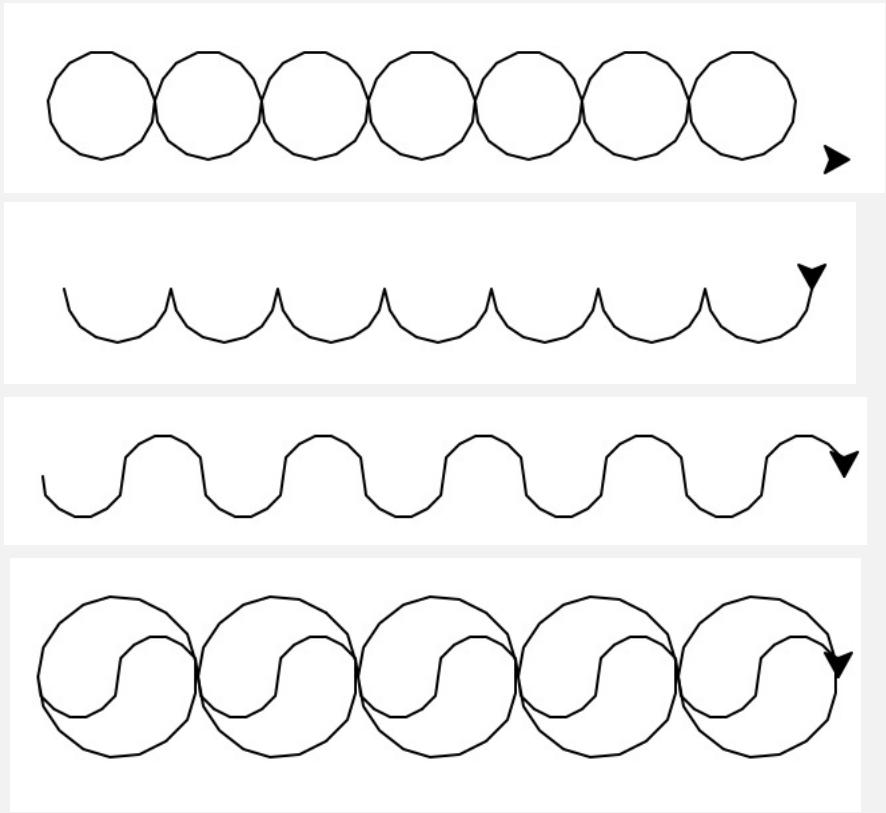
```
1  from turtle import *
2  for x in range (5):
3      circle(20)
4      penup()
5      forward(50)
6      pendown()
```

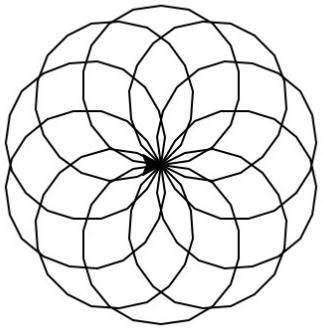


Για μεγαλύτερη ταχύτητα εκτέλεσης γράψτε την εντολή **speed(0)**

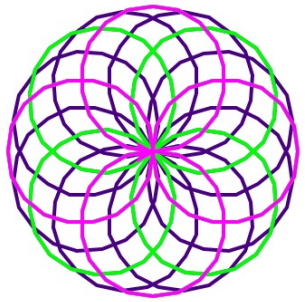
Δραστηριότητα 3

Χρησιμοποιώντας την **for** και την **circle** δημιουργήστε τα παρακάτω σχήματα:

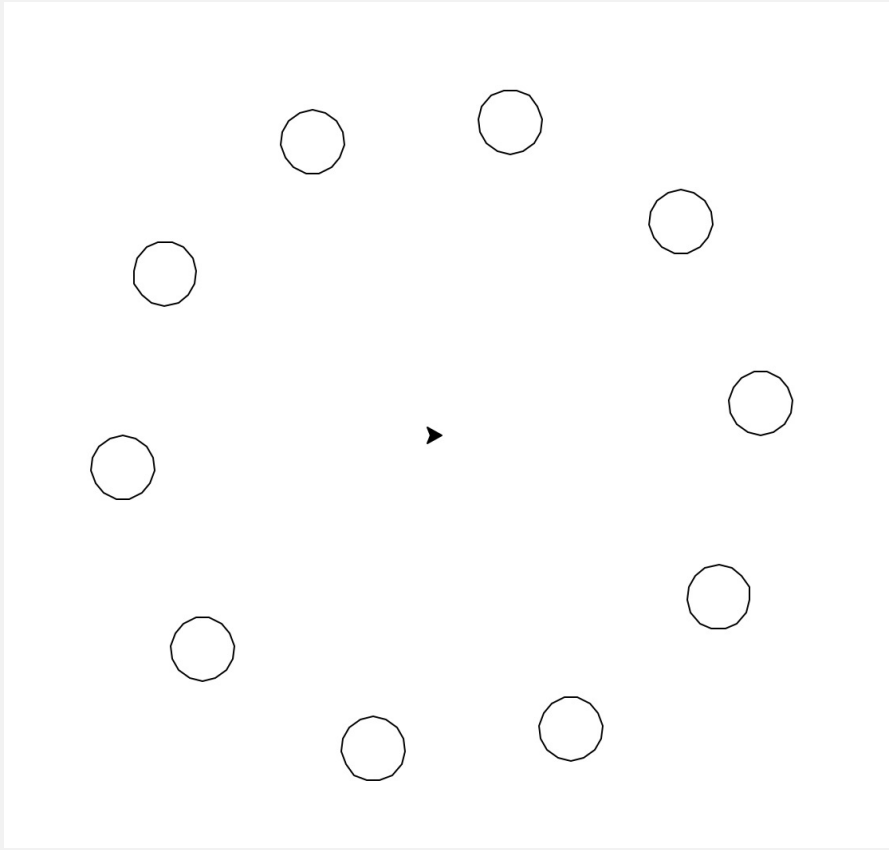




10 κύκλοι



16 – 8 – 4 κύκλοι



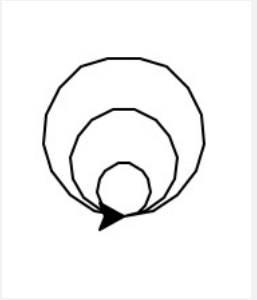
Σχήματα και μεταβλητές

Αν θέλουμε μπορούμε να χρησιμοποιήσουμε και **μεταβλητές** ως παραμέτρους των εντολών κίνησης της χελώνας. Για παράδειγμα:

```
1  from turtle import *
2  x=10
3  circle(x)
```

Θα μπορούσαμε να αλλάξουμε τον προηγούμενο κώδικα ώστε να σχεδιάσουμε **3 κύκλους**, τον καθένα κατά **10 px μεγαλύτερο από τον προηγούμενο**. Ο κώδικας σε αυτή την περίπτωση γίνεται:

```
1  from turtle import *
2  x=10          #αρχικοποιώ τη μεταβλητή x στην τιμή 10
3  circle(x)     #σχεδιάζω έναν κύκλο με ακτίνα x(δηλ 10)
4  x=x+10        #αυξάνω το x κατά 10
5  circle(x)     #σχεδιάζω νέο κύκλο με ακτίνα x(δηλ 20)
6  x=x+10        #αυξάνω το x κατά 10
7  circle(x)     #σχεδιάζω νέο κύκλο με ακτίνα x(δηλ 30)
```

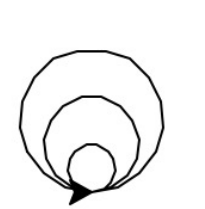


Οι προτάσεις που βλέπετε να ακολουθούν τα σύμβολο **#** και είναι με γκρι χρώμα, ονομάζονται **σχόλια** και δεν επηρεάζουν καθόλου το πρόγραμμα.

Σχήματα και μεταβλητές

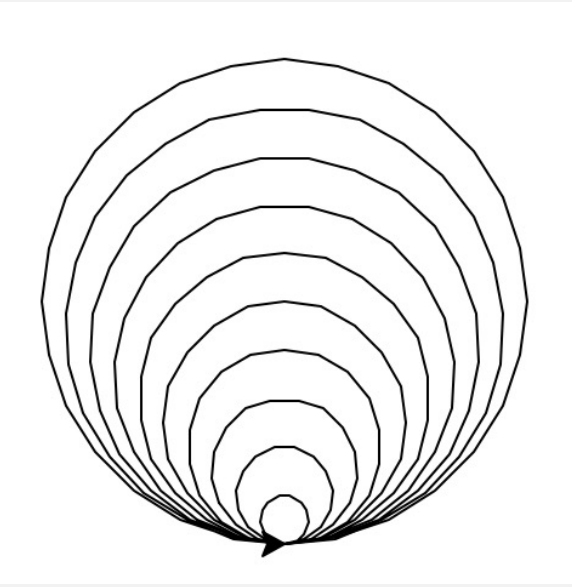
Ο προηγούμενος κώδικας μπορεί να γραφεί εναλλακτικά και με χρήση της **δομής επανάληψης** (**for**). Σε αυτή την περίπτωση ο κώδικας γίνεται:

```
1  from turtle import *
2  x=10                    #αρχικοποιώ τη μεταβλητή x στην τιμή 10
3  for i in range(3):      #επαναλαμβάνω 3 φορές (προσοχή χρησιμοποιώ άλλη μεταβλητή)
4      circle(x)           #σχεδιάζω κύκλο με ακτίνα x
5      x=x+10              #αυξάνω το x κατά 10
```



Απλά αλλάζοντας τον αριθμό των επαναλήψεων από 3 σε έναν μεγαλύτερο αριθμό, πχ **10**, θα έχω ένα πολύ πιο εντυπωσιακό σχήμα.

```
1  from turtle import *
2  x=10
3  for i in range(10):
4      circle(x)
5      x=x+10
```



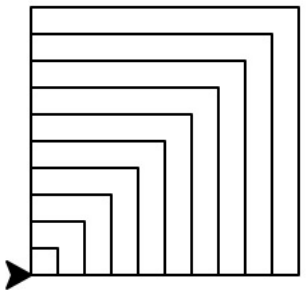
Πολλά τετράγωνα αυξανόμενου μεγέθους

Για να κάνουμε ένα τετράγωνο με πλευρά x αρκεί να γράψουμε τον παρακάτω κώδικα.

```
for y in range(4):
    forward(x)
    left(90)
```

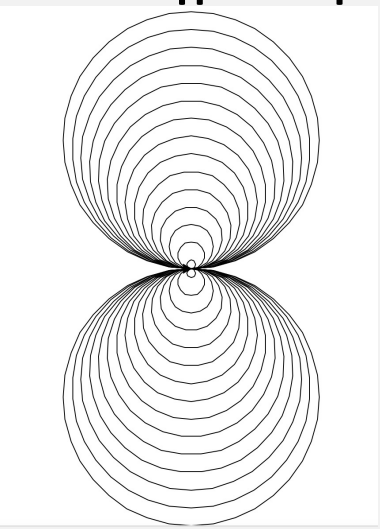
Επομένως αν θέλουμε να κάνουμε ένα αντίστοιχο σχήμα με αυτό της προηγούμενης άσκησης απλά αντί για 10 κύκλους να σχεδιάσουμε 10 αυξανόμενου μεγέθους τετράγωνα αρκεί να αντικαταστήσουμε την εντολή circle(x) με τον παραπάνω κώδικα. Σε αυτή την περίπτωση ο κώδικας γίνεται:

```
1  from turtle import *
2  x=10
3  for i in range(10):
4      for y in range(4):
5          forward(x)
6          left(90)
7      x=x+10
```

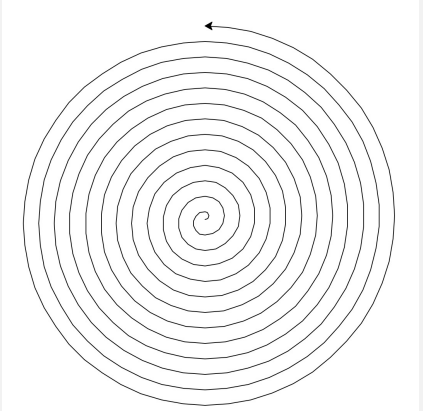


Δραστηριότητα 4

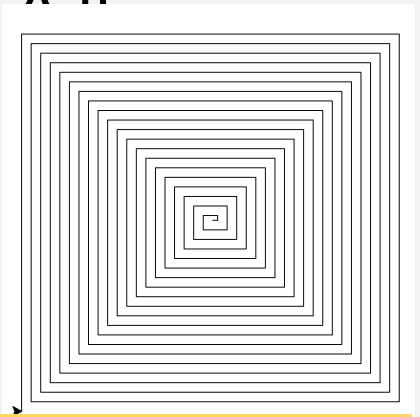
Χρησιμοποιώντας τα δύο προηγούμενα παραδείγματα προσπαθήστε να δημιουργήσετε τα παρακάτω σχήματα:



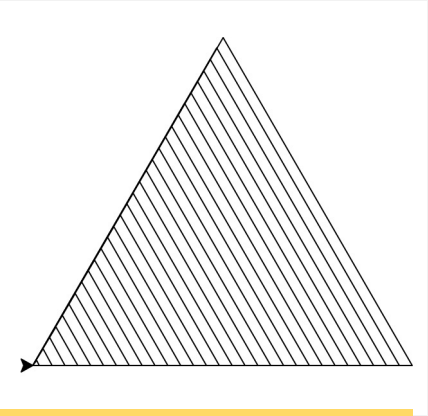
A: **5** E: **15** B:**10**



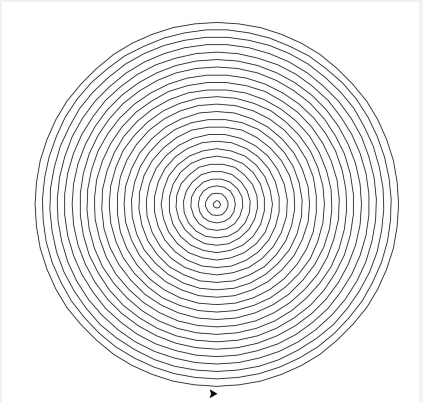
A: **5** E: **25** B:**10**



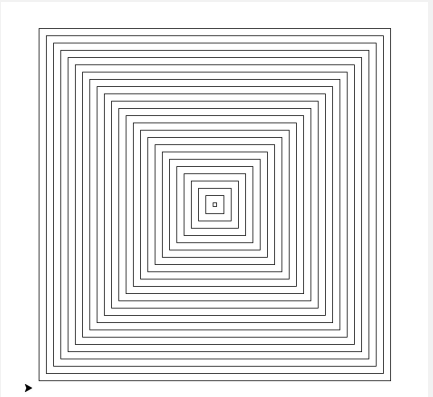
A: **5** E: **40** B:**10**



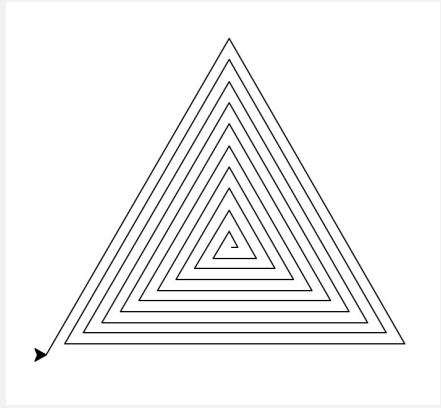
A: **5** E: **30** B:**10**



A: **5** E: **25** B:**10**



A: **5** E: **25** B:**20**



A: **5** E: **30** B:**10**

A: Αρχικοποίηση
E: Αριθμός Επαναλήψεων
B: Βήμα (Αύξηση μεταβλητής)