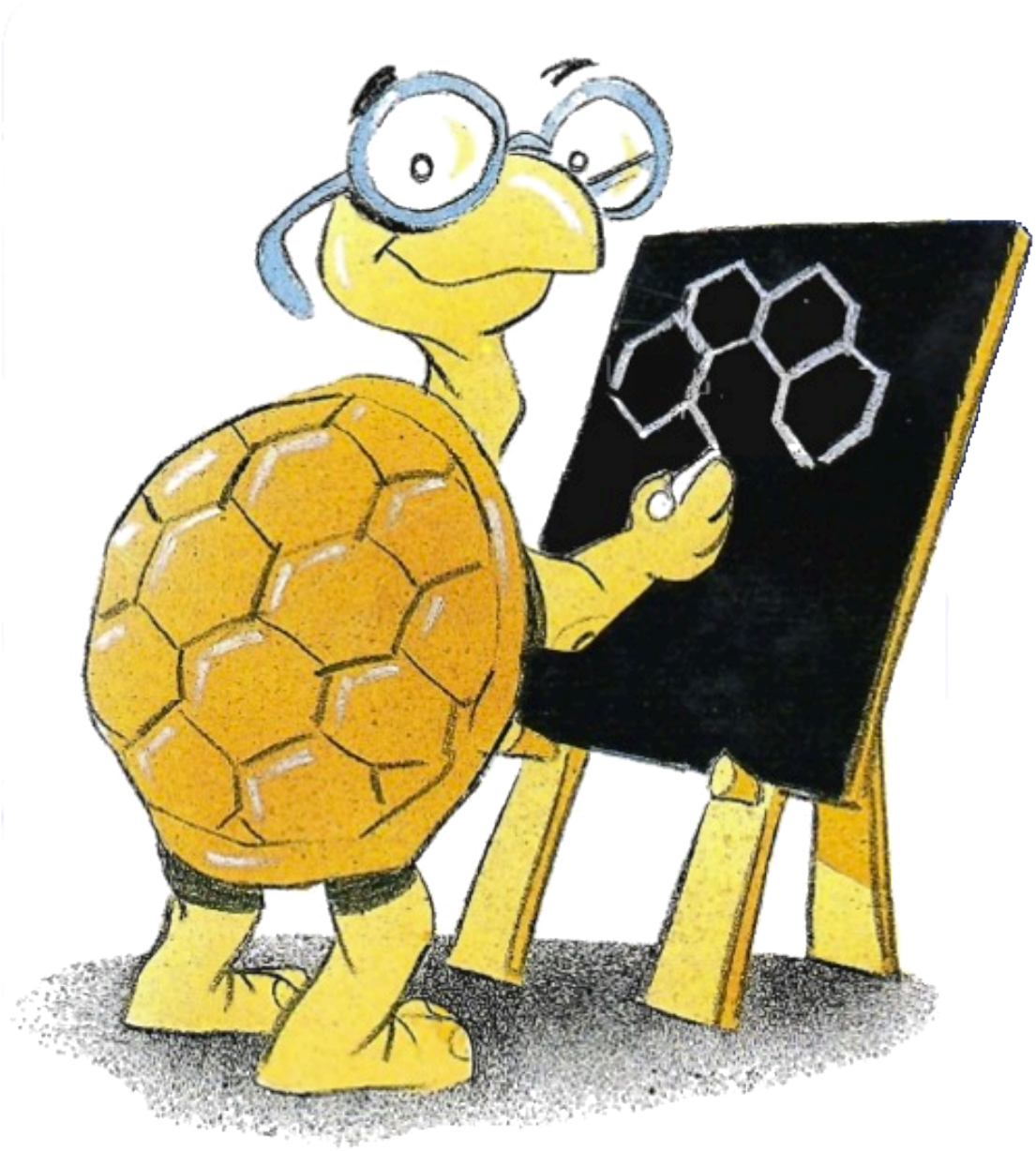
Python Turtle
Armenian/English Reference 1.1e
Հայերեն/Անգլերեն Օժանդակ

Full Python Turtle documentation available at docs.python.org
Turtle Art and Puzzles from "Turtle Confusion" by Dr. Barry Newell
Translated and compiled by Gagik Movsisyan



import turtle

Ներմուծել “Python Turtle” մոդուլը: Անհրաժեշտ է անել օգտագործելուց առաջ:

Import the “Python Turtle” module. This is necessary before using the module.

myturtle = turtle.Turtle()

Ստեղծել նոր կրիա նշված անունով: myturtleի փոխարեն կարողեք դնել ձեր ցանկացած անունը:

Create a new turtle with the given name. You may name the turtle anything you like as long as it is not a keyword.

myturtle.shape(*name*)

Փոխել կրիայի պատկերը:

Set turtle shape to shape with given *name*. Initially there are the following polygon shapes: “arrow”, “turtle”, “circle”, “square”, “triangle”, “classic”.

Parameters
Պարամետրեր

- **name** (պատկերի անունը) – a string which is a valid shapename (բառ)

կրիա: **“turtle”** կլոր: **“circle”** քառակուսի: **“square”**
եռանկյունի: **“triangle”** սլաք: **“arrow”**

```
>>> myturtle.shape("turtle")
>>> myturtle.shape("circle")
```

myturtle.**color**(*color*)

Փոխել կրիայի և գրիչի գույնը:

Return or set pencolor and fillcolor.

Parameters
Պարամետրեր

- **color** (գույնը) – a colorstring or a numeric color tuple (բառ կամ գույնի կոդը)

```
>>> myturtle.color("red") # ուրվագիծ և լցում: կարմիր
>>> myturtle.color("red", "green") # ուրվագից: կարմիր, լցում: կանաչ
```

myturtle.**fillcolor**(*color*)

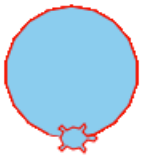
Փոխել լցնելու գույնը:

Return or set fillcolor.

Parameters
Պարամետրեր

- **color** (գույնը) – a colorstring or a numeric color tuple (բառ կամ գույնի կոդը)

```
>>> myturtle.color("red")
>>> myturtle.fillcolor("skyBlue")
```



myturtle.**pensize**(*width*)

= `myturtle.width`(*width=None*)

Փոխել գծի հաստությունը:

Set the line thickness to *width* or return it.

Parameters
Պարամետրեր

- **width** (հաստությունը) – a number (թիվ)

```
>>> myturtle.pensize(10)
```

myturtle.**forward**(*distance*)

= myturtle.fd(*distance*)

Առաջ շարժել կրիային նշված հեռավորությունով: (distance = հեռավորություն)

Move the turtle forward by the specified *distance*, in the direction the turtle is headed.

Parameters
Պարամետրեր

- **distance** (հեռավորությունը) – a number (թիվ)

```
>>> myturtle.forward(25)
```

myturtle.**back**(*distance*)

= myturtle.bk(*distance*)
= myturtle.backward(*distance*)

Ետ շարժել կրիային նշված հեռավորությունով: (distance = հեռավորություն)

Move the turtle backward by *distance*, opposite to the direction the turtle is headed. Do not change the turtle's heading.

Parameters
Պարամետրեր

- **distance** (հեռավորությունը) – a number (թիվ)

```
>>> myturtle.backward(30)
```

myturtle.**goto**(*x*, *y*)

= myturtle.setpos(*x*, *y*)
= myturtle.setposition(*x*, *y*)

Կրիային տեղափոխել նշված կոորդինատներով:

Move turtle to an absolute position. If the pen is down, draw line. Do not change the turtle's orientation.

Parameters
Պարամետրեր

- **x** – a number (թիվ)
- **y** – a number or `None` (թիվ կամ ոչինչ)

```
>>> myturtle.goto(60,30)
```

myturtle.**right**(*angle*)

= myturtle.rt(*angle*)

Աջ շրջել կրիային նշված անկյամբ:

Turn turtle right by *angle* units. (Units are by default degrees, but can be set via the `degrees()` and `radians()` functions.) Angle orientation depends on the turtle mode, see `mode()`.

Parameters
Պարամետրեր

- **angle** (անկյունաչափ) – a number (integer or float) (թիվ)

```
>>> myturtle.right(45)
```

myturtle.**left**(*angle*)

= myturtle.lt(*angle*)

Ձախ շրջել կրիային նշված անկյամբ:

Turn turtle left by *angle* units.

Parameters
Պարամետրեր

- **angle** (անկյունաչափ) – a number (integer or float) (թիվ)

```
>>> myturtle.left(45)
```

myturtle.**speed**(*speed*)

Փոխել կրիայի արագությունը:

Set the turtle's speed to an integer value in the range 0..10. If no argument is given, return current speed.

- "fastest" (Ամենաարագ): 0
- "fast" (Արագ): 10
- "normal" (Նորմալ): 6
- "slow" (Դանդաղ): 3
- "slowest" (Ամենադանդաղ): 1

Parameters
Պարամետրեր

- **speed** (արագությունը) – an integer in the range 0..10 or a speedstring (see below) (թիվ 0...10)

```
>>> myturtle.speed(9)
```

myturtle.**circle**(*radius, extent=None, steps=None*)

Նկարել շրջան նշված շառավղով:

Draw a circle with given *radius*. The center is *radius* units left of the turtle; *extent* – an angle – determines which part of the circle is drawn. If *extent* is not given, draw the entire circle. If *extent* is not a full circle, one endpoint of the arc is the current pen position. Draw the arc in counterclockwise direction if *radius* is positive, otherwise in clockwise direction. Finally the direction of the turtle is changed by the amount of *extent*.

Parameters
Պարամետրեր

- **radius** (շառավիղ) – a number (թիվ)
- **extent** (աստիճան) – a number(or `None`)(թիվ կամ ոչինչ)
- **steps** (քայլեր) – an integer (or `None`) (թիվ կամ ոչինչ)

```
>>> myturtle.circle(50)  # շրջան 50 շառավղով
>>> myturtle.circle(120, 180)  # կիսաշրջան 120 շառավղով
```

myturtle.**dot**(*size=None, *color*)

Նկարել կետ նշված տրամագծով և գույնով:

Draw a circular dot with diameter *size*, using *color*. If *size* is not given, the maximum of pensize+4 and 2*pensize is used.

Parameters
Պարամետրեր

- **size** (չափսը) – an integer >= 1 (if given)
  (թիվ >= 1 կամ ոչինչ)
- **color** (գույնը) – a colorstring or a numeric color tuple ը)
  (բառ կամ գույնի կոդը)

```
>>> myturtle.dot(20, "blue");
```

`myturtle.stamp()`

Դրոշմել կրիայի նկարի վրա:

Stamp a copy of the turtle shape onto the canvas at the current turtle position. Return a stamp_id for that stamp, which can be used to delete it by calling `clearstamp(stamp_id)`.

`myturtle.pendown()`

`= myturtle.pd()`
`= myturtle.down()`

Գրիչը դնել ցած որ շարժվելուց նկարվի:

Pull the pen down – drawing when moving.

`myturtle.penup()`

`= myturtle.pu()`
`= myturtle.up()`

Գրիչը վերցնել որ շարժվելուց չ՝նկարվի:

Pull the pen up – no drawing when moving.

myturtle.**begin_fill()**

Կատարել լցվող պատկեր նկարելուց առաջ:

Call just before drawing a shape to be filled. Equivalent to `fill(True)`.

myturtle.**end_fill()**

Կատարել լցվող պատկեր նկարելուց հետո:

Fill the shape drawn after the last call to `begin_fill()`

```
>>> myturtle.color("red")
>>> myturtle.fill_color("lightBlue")
>>> myturtle.begin_fill()
>>> myturtle.circle(30)
>>> myturtle.end_fill()
```

mycanvas = turtle.**Screen**()

Ստեղծել կտավ նշված անունով:

mycanvas.**setup**(*width, height, startx, starty*)

Փոխել կտավի չափսը և դիրքը:

Parameters
Պարամետրեր

- **width** (լայնություն) – (թիվ)
- **height** (բարձրություն) – (թիվ)
- **startx** (սկզբնական x կոորդինատ) – (թիվ)
- **starty** (սկզբնական y կոորդինատ) – (թիվ)

mycanvas.**bgcolor**(*color*)

Փոխել կտավի գույնը:

Parameters
Պարամետրեր

- **color** (գույնը) – a colorstring or a numeric color tuple ը) (բառ կամ գույնի կոդը)

Python Reference

Loops – Կրկնվող գործողություններ

For loop

Example 1

code - կոդ

```
for i in range(5):
  print 'hello'
```

քանի անգամ կրկնել

կրկնվող գործողությունը

output – արդյունքը

```
hello
hello
hello
hello
hello
```

Example 2

code - կոդ

```
for i in range(5):
  print 'hello'
  print i
```

քանի անգամ կրկնել

կրկնվող գործողությունը

output – արդյունքը

```
hello
0
hello
1
hello
2
hello
3
hello
4
```

While loop

Example 1

code – կոդը

```
x = 1
while x < 5:
   print x
   x = x + 1
```

կ'կրկնվի մինչդեռ սա ճիշտ է

կրկնվող գործողությունը

output – արդյունքը

```
1
2
3
4
```

Example 2

code - կոդը

```
while True:
   print 'hello'
```

կ'կրկնվի մինչդեռ սա ճիշտ է

կրկնվող գործողությունը

output – արդյունքը

```
hello
hello
hello
hello
hello
hello
hello
hello
.
.
.
```

կ'շարունակի տպել 'hello' ընդմիշտ

Conditionals

If statements

Example 1

code - կոդը

```
x = 16
if x < 10:
  print x, 'is less than 10'
else:
  print x, 'is not less than 10'
```

սա կ'կատարվի եթե սա ճիշտ է

եթե ճիշտ չէ, սա կ'կատարվի

output – արդյունքը

```
16 is not less than 10
```

Example 2

code - կոդը

```
x = 16
if x < 10:
  print x, 'is less than 10'
elif x < 20:
  print x, 'is between 10 and 20'
else:
  print x, 'is greater than 20'
```

output – արդյունքը

```
16 is between 10 and 20
```