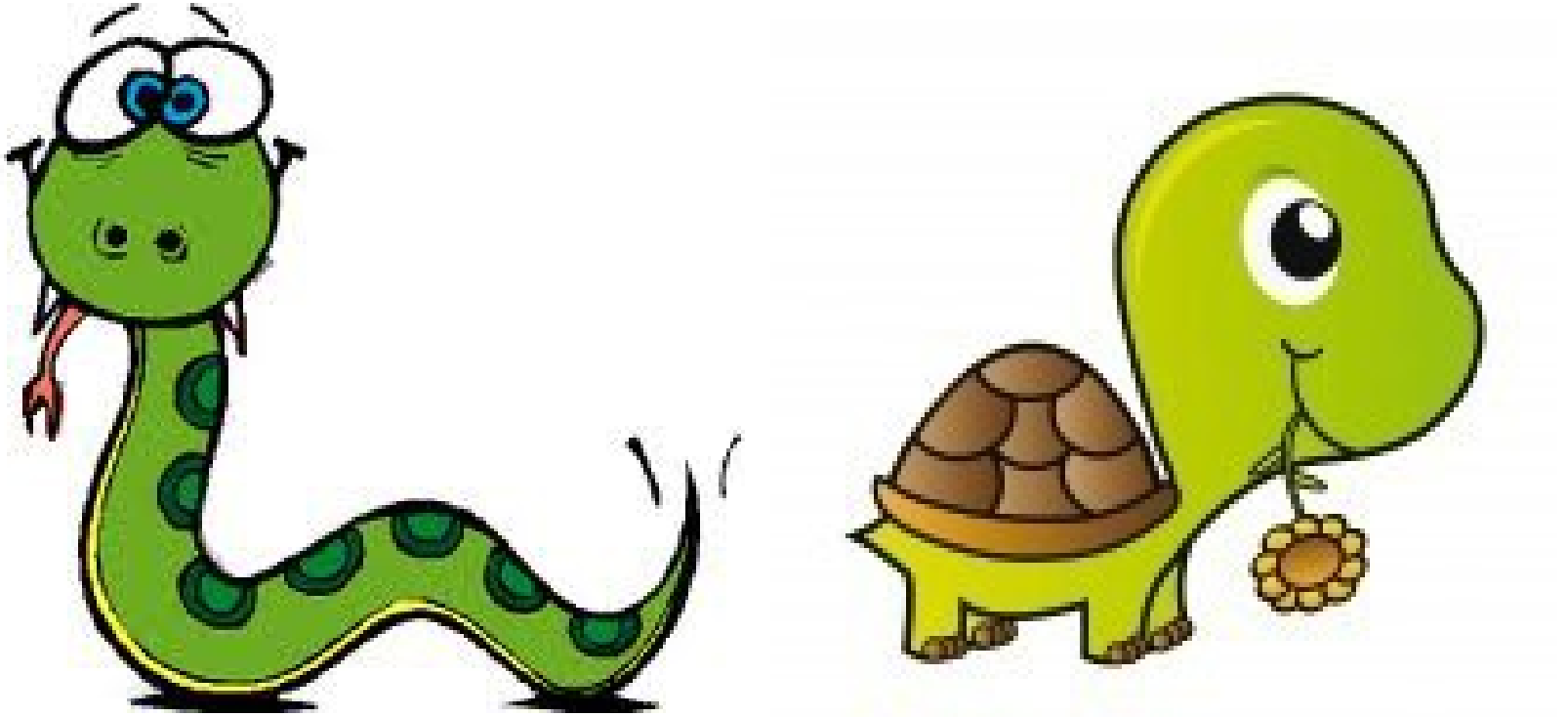
Python
Il modulo Turtle



sophia.danesino@peano.it

4/12/2013

Modulo per biennio

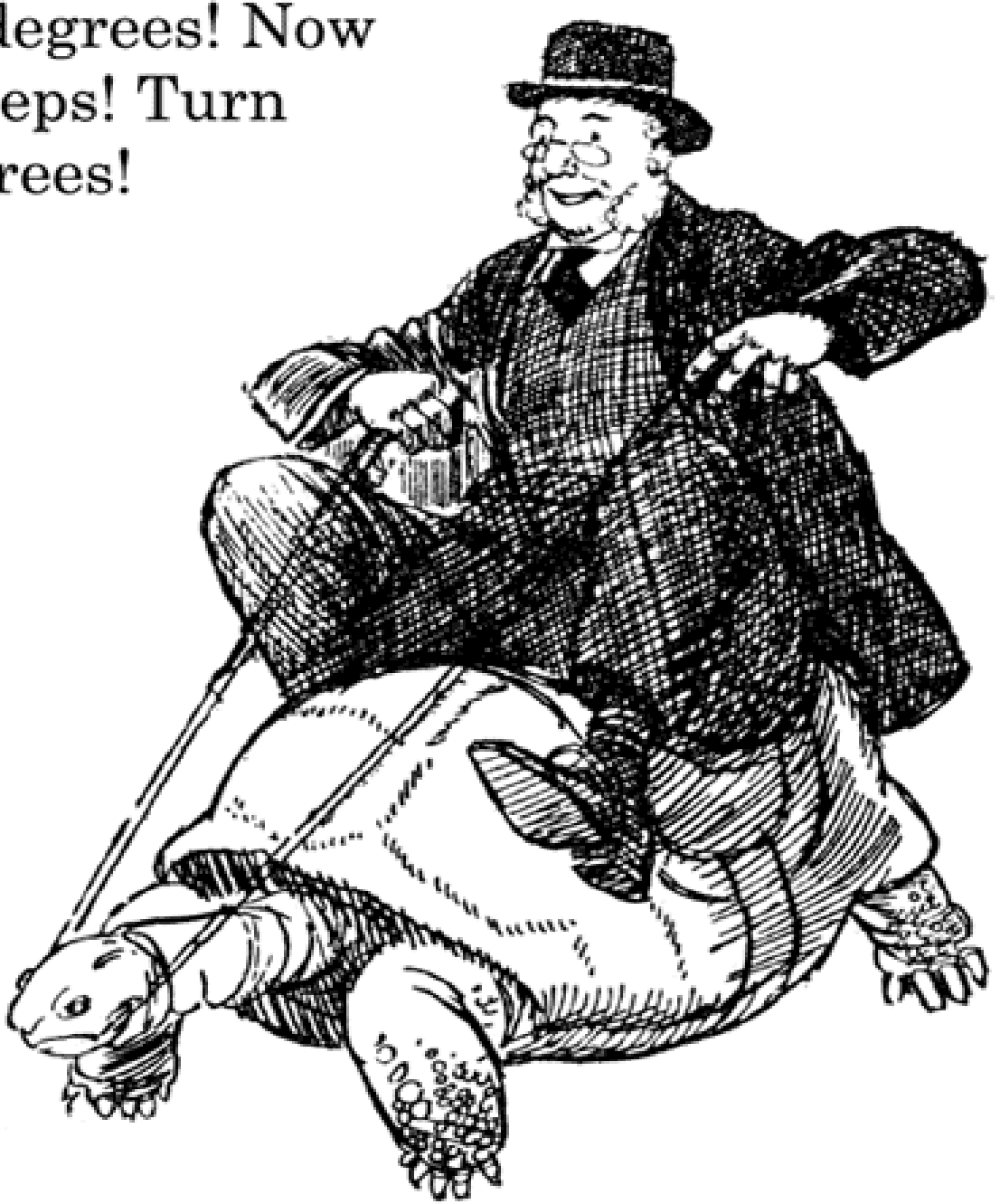
- Orientamento nello spazio
- Cicli
- Funzioni
- Ricorsione
- OOP

Dove trovo la documentazione?

http://orientamento.educ.di.unito.it/

Informatica per le scuole secondarie
Workshop Teachers for Teachers
T4T 2013

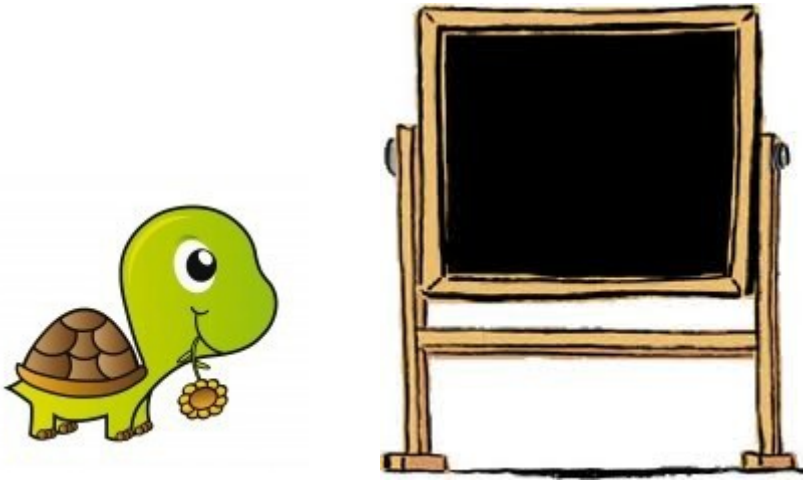
Forward 20 steps! Now turn right, by 45 degrees! Now go back 40 steps! Turn right, 90 degrees!



Portable Python

http://www.portablepython.com/

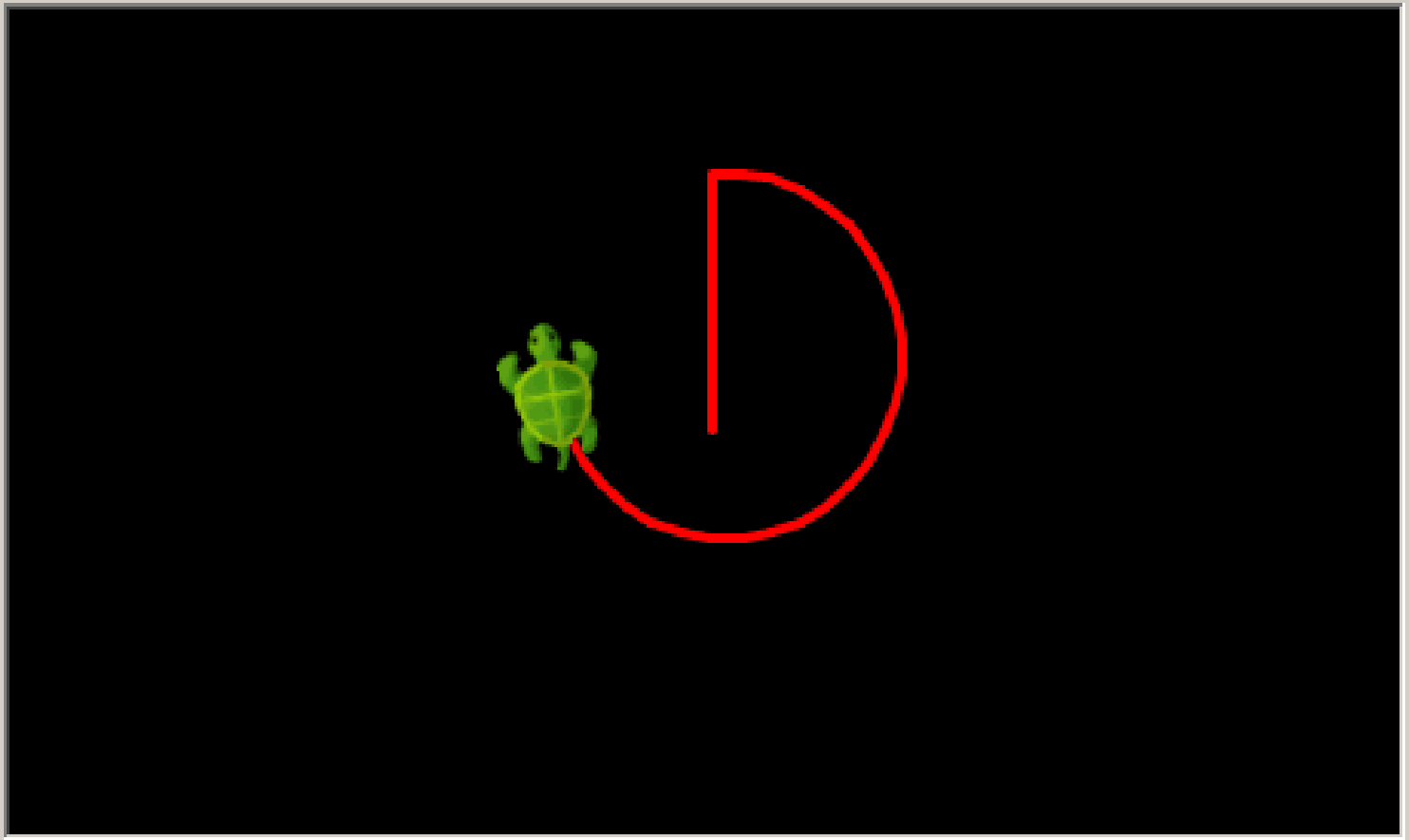
Turtle Python



- http://pythonturtle.org/
- *A learning environment for Python suitable for beginners and children, inspired by Logo.*

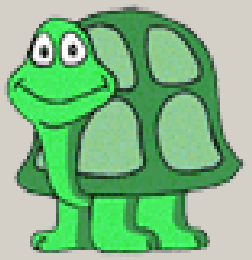
PythonTurtle

File Help

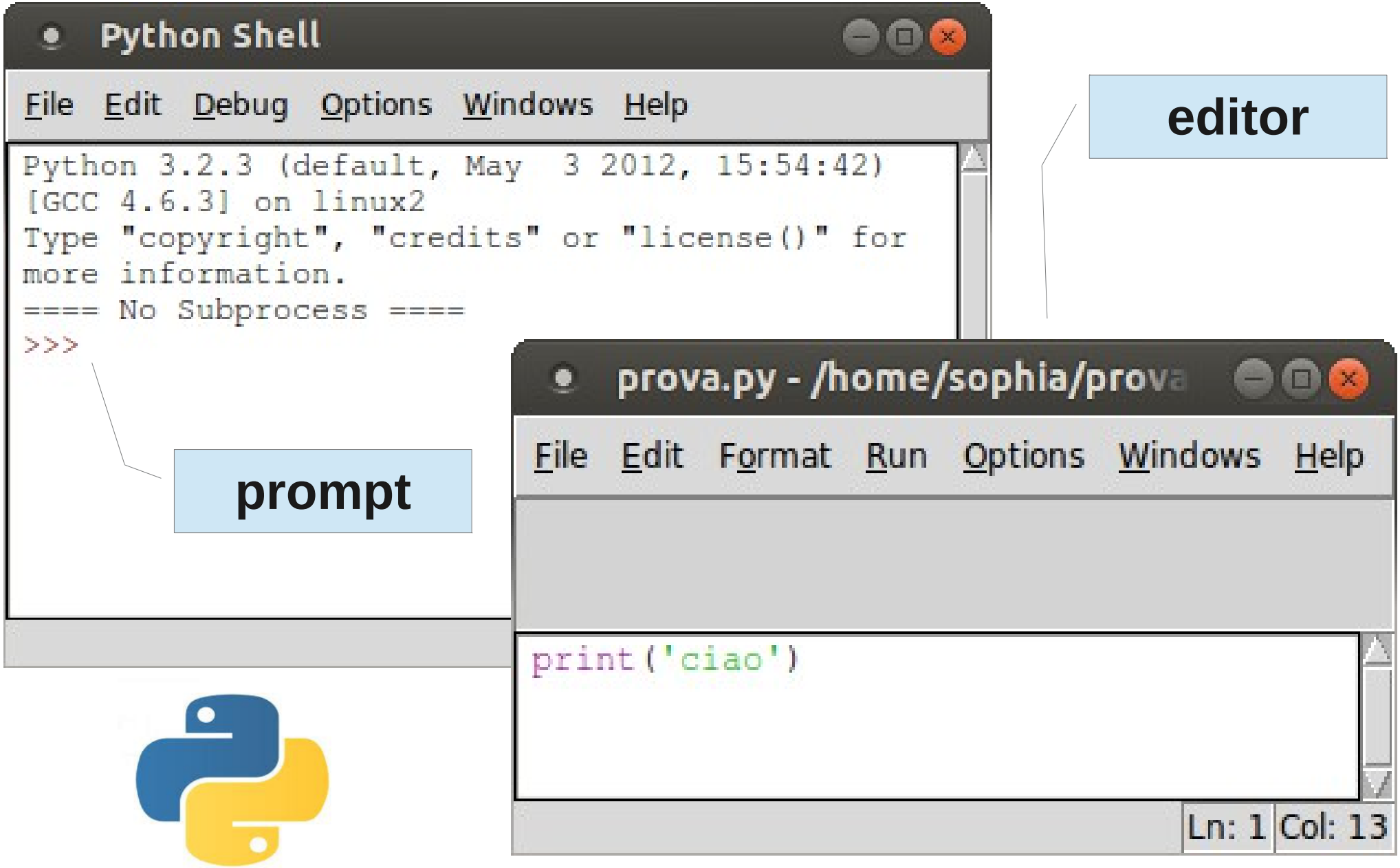


```
>>> go(80)
>>> turn(90)
>>> for i in range(26):
...     go(10)
...     turn(10)
...
>>>
```

Teach me



IDLE Integrated Development Environment



COMANDI PER ESECUZIONE

PyScripter - modulo1*

File Modifica Cerca Visualizza Progetto Esegui Strumenti Aiuto

File Explorer

Computer
Acer (C:)

FILE EXPLORER

```
#-------------------------------------------------------------------------------
# Name:        modulo1
# Purpose:
#
# Author:      Sofia
#
# Created:     26/09/2012
# Copyright:   (c) Sofia 2012
# Licence:     <your licence>
#-------------------------------------------------------------------------------
#!/usr/bin/env python

print("ciao")
```

EDITOR CODICE

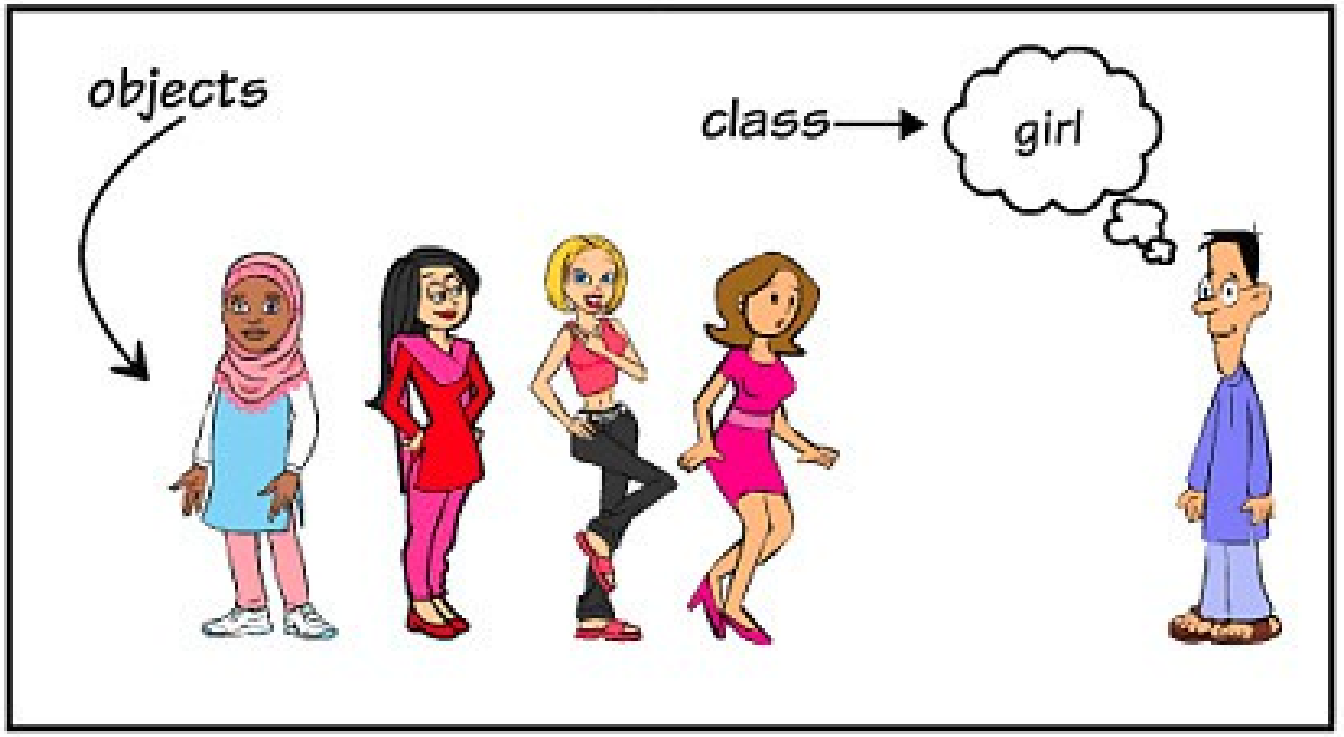
File ... | Expl... | Expl... | modulo1

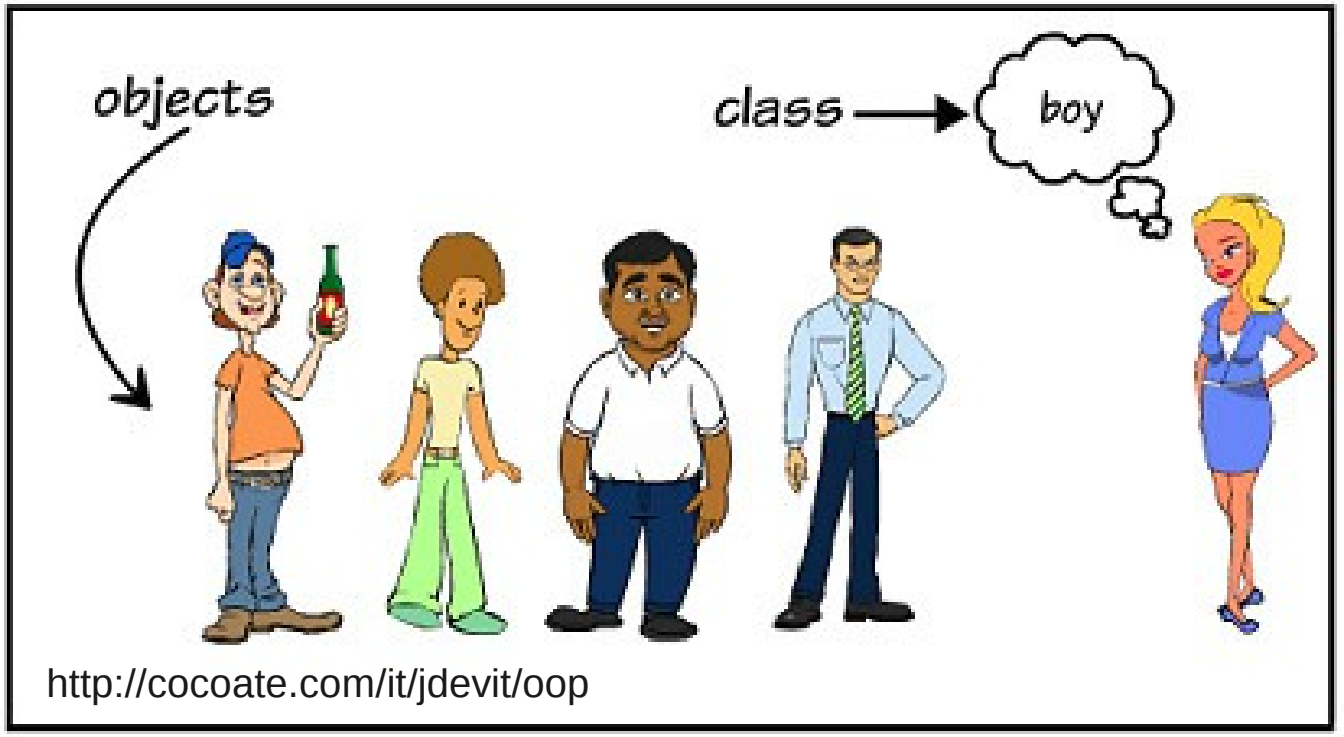
Interprete Python

```
*** Python 3.2.1 (default, Jul 10 2011, 21:51:15) [MSC v.1500 32 bit (Intel)] on win32. ***
*** L'engine Python Remote  è attivo ***
>>>
*** Interprete Remoto Reinizializzato ***
>>>
ciao
>>>
```

SHELL (OUTPUT)

Stack chiamate | Variabili | Osservazioni | Breakpoint | Output | Messaggi | Interprete Python





http://cocoate.com/it/jdevit/oop

OOP

Una **classe** è un tipo di un oggetto

Un oggetto è un'**istanza** di una classe

Un'istanza ha proprietà (o **attributi**) e comportamenti (o **metodi**) definiti dalla classe.

1. Importare il modulo

from turtle import Turtle

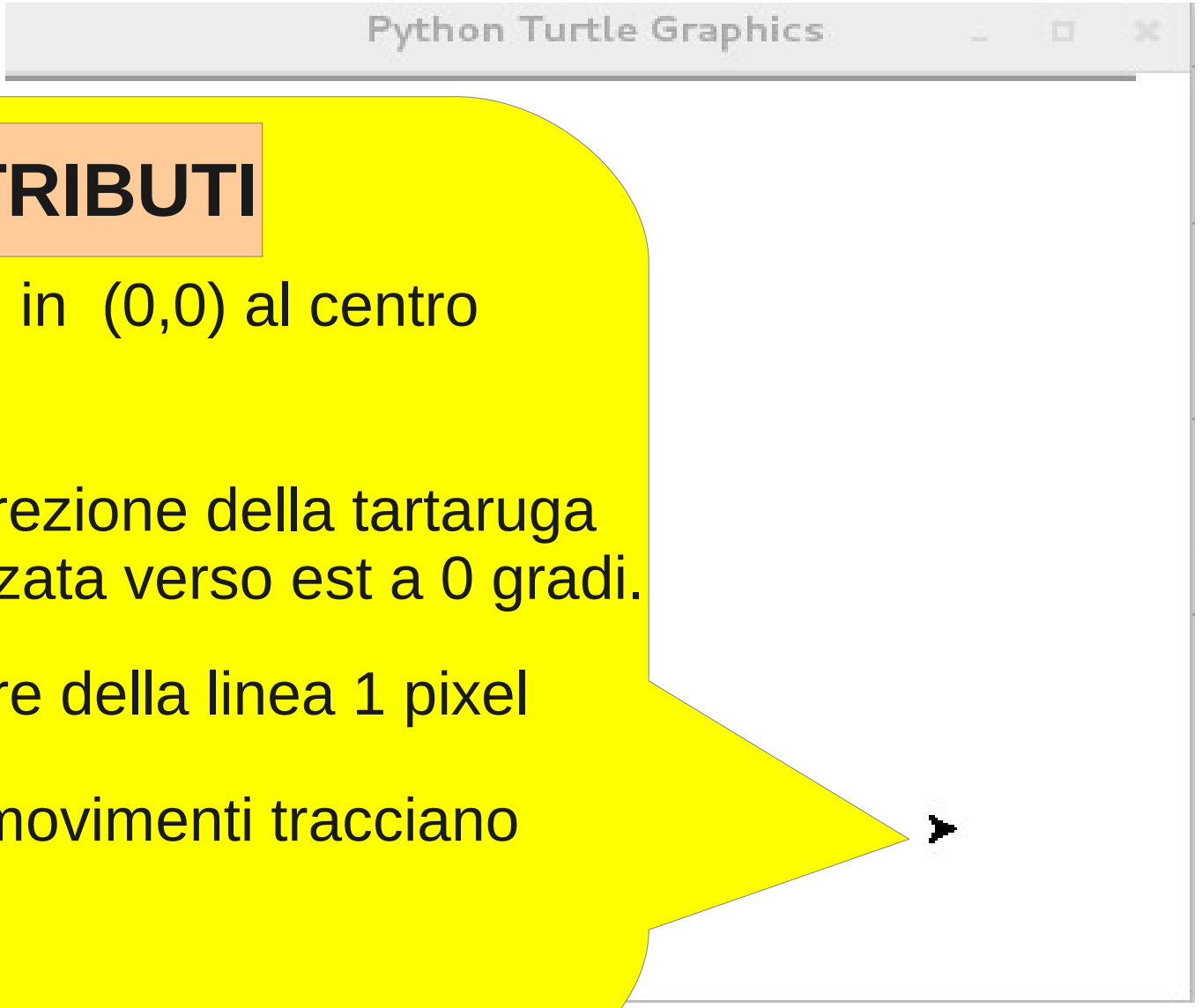
2. Creare un oggetto Turtle

ruga = Turtle()

3. Richiamare i metodi dell'oggetto

ruga.forward(100)

```
>>>
>>> from turtle import Turtle
>>> ruga = Turtle()
>>>
```



ATTRIBUTI

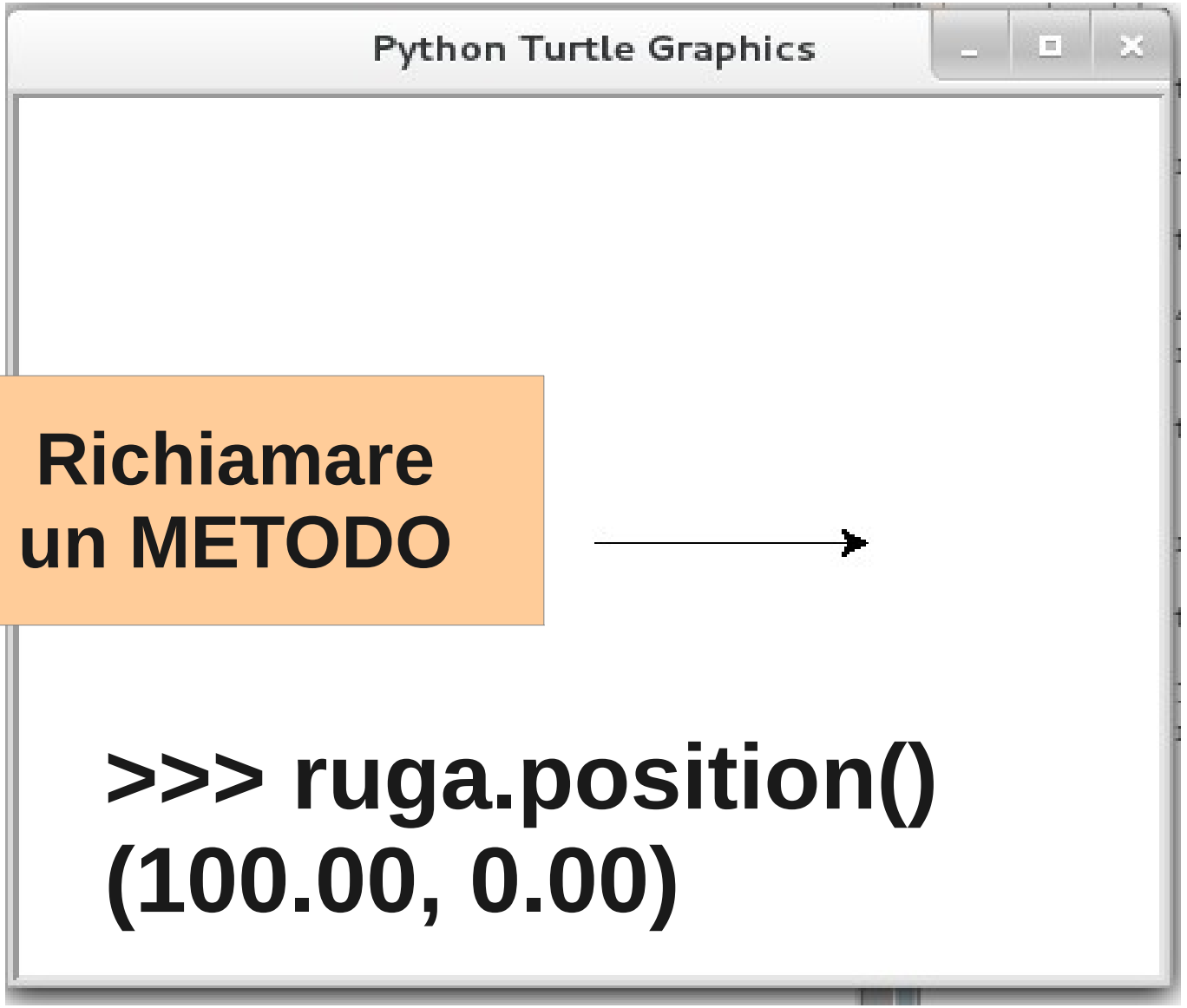
HOME: l'origine, in (0,0) al centro della finestra.

HEADING: la direzione della tartaruga in gradi. Inizializzata verso est a 0 gradi.

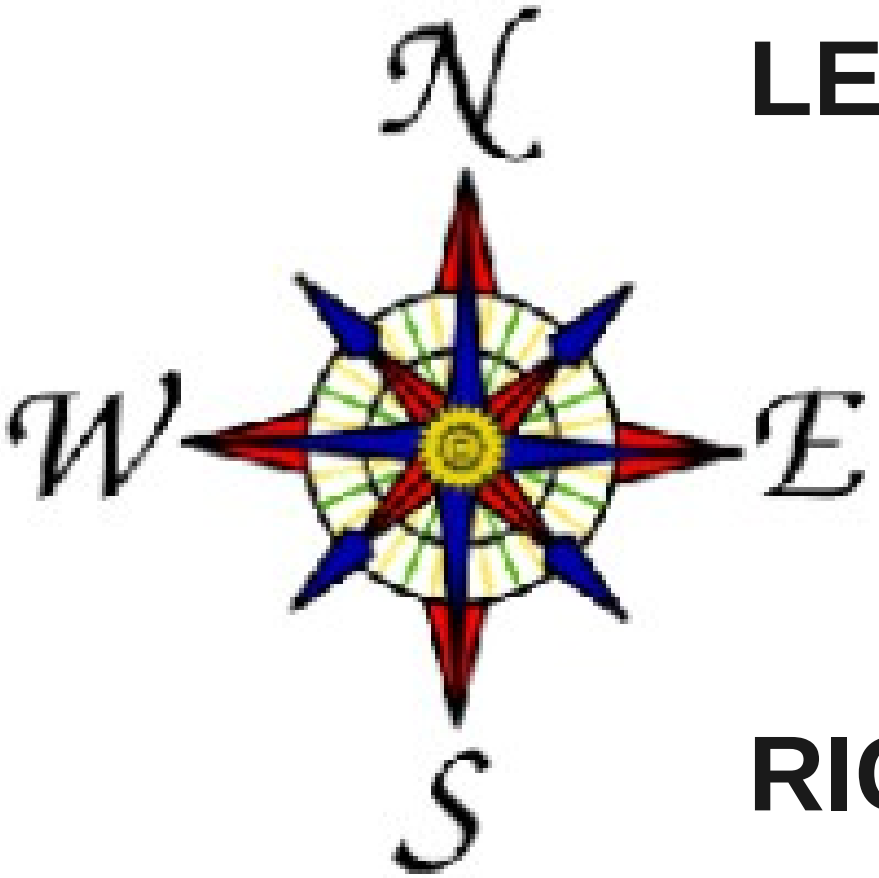
WIDTH: spessore della linea 1 pixel

DOWN: vero (i movimenti tracciano una linea)

>>> ruga.forward(100)



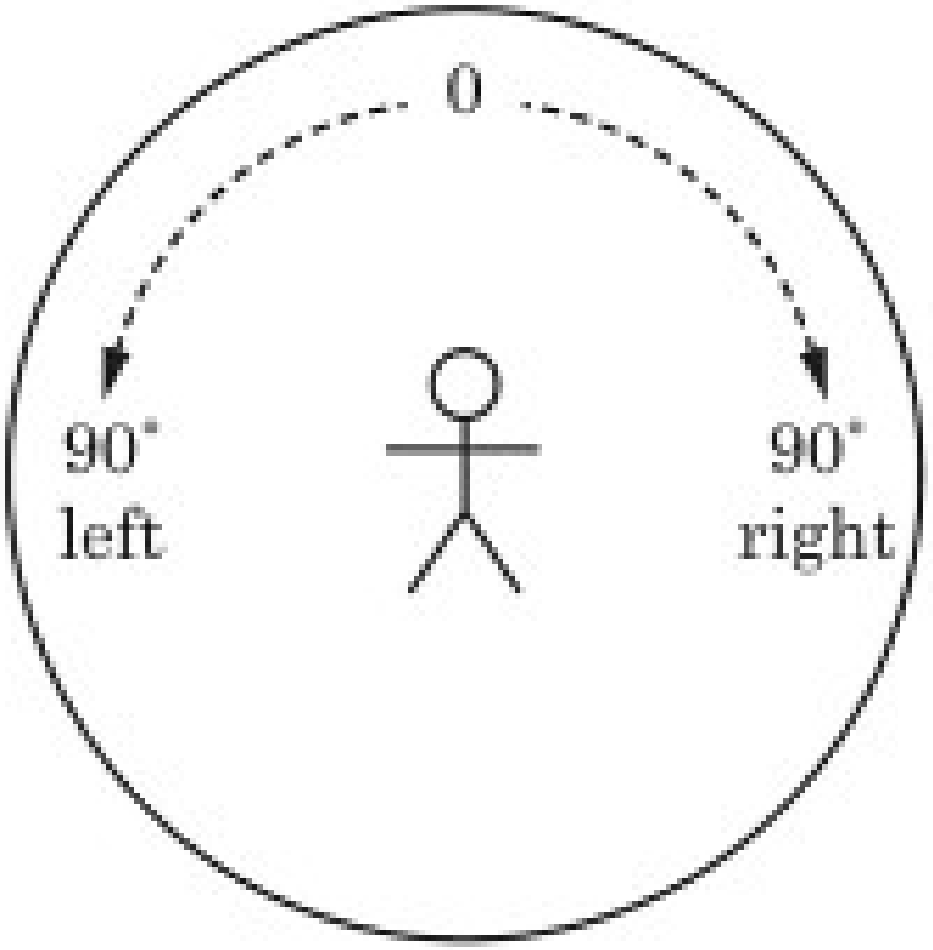
Coordinate della tartaruga

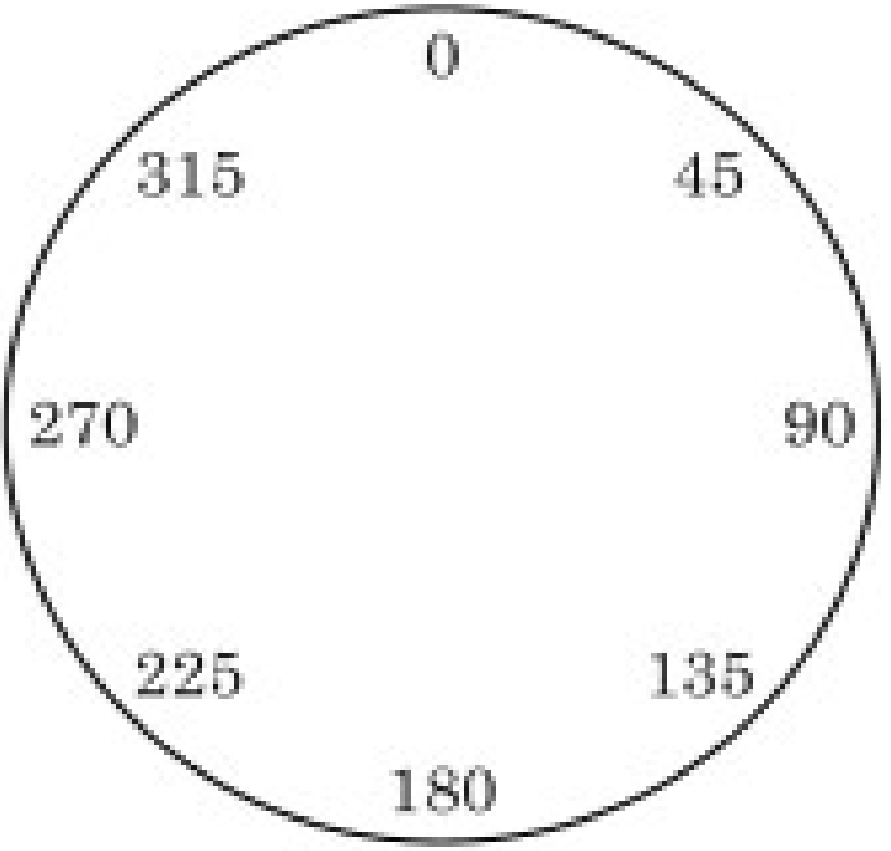


LEFT

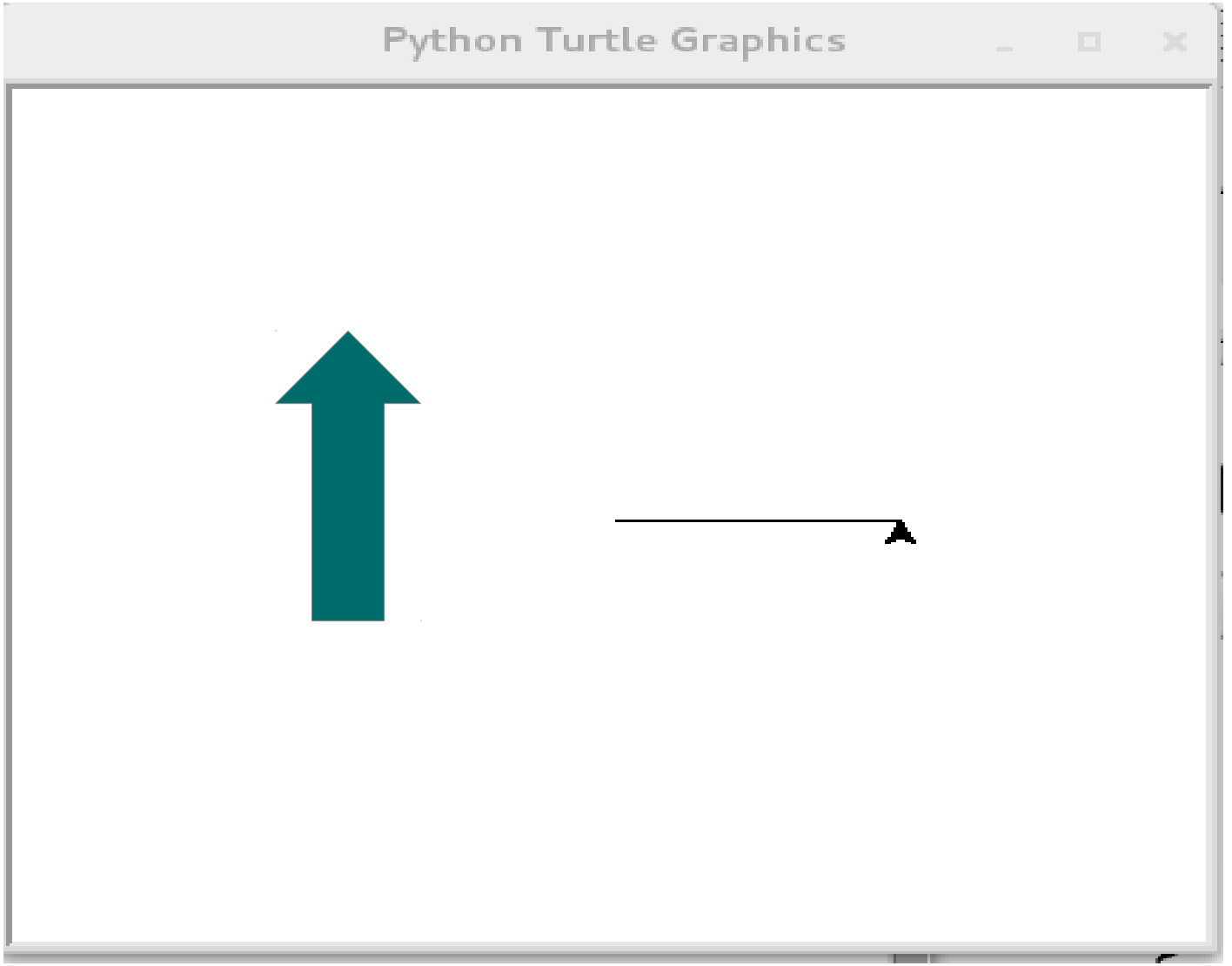
ALLA CREAZIONE

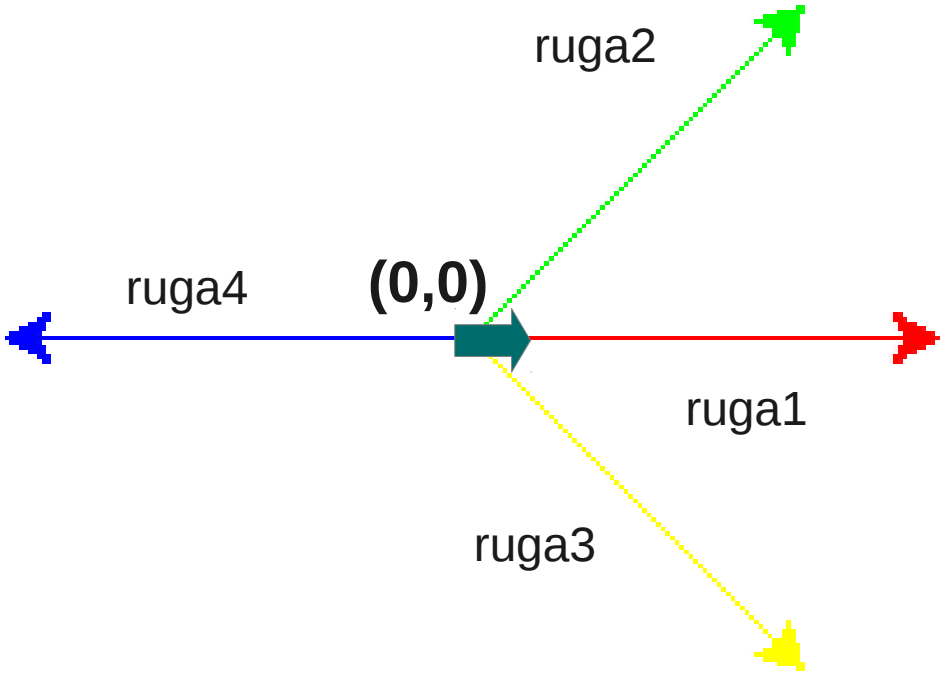
RIGHT


0
90°
left
90°
right


0
315
45
270
90
225
135
180

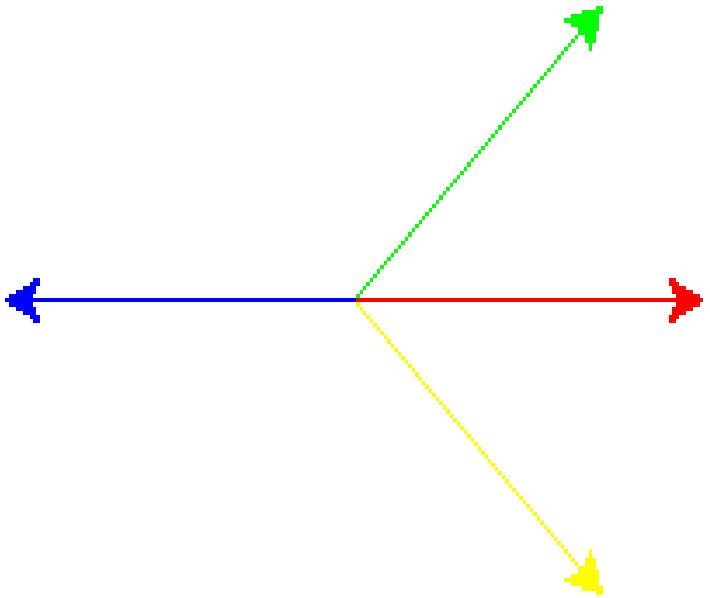
```
>>> ruga.left(90)
>>> ruga.heading()
90.0
```

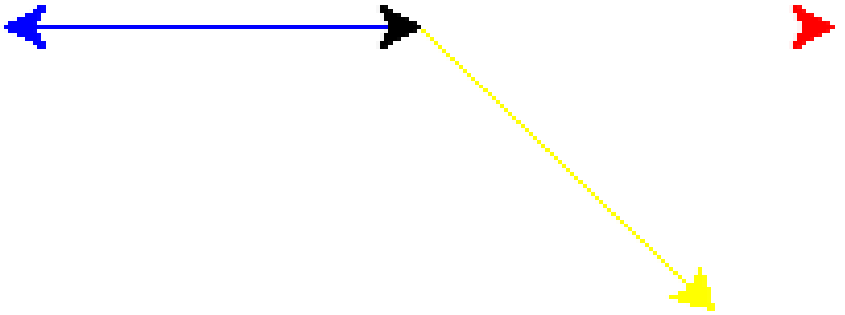




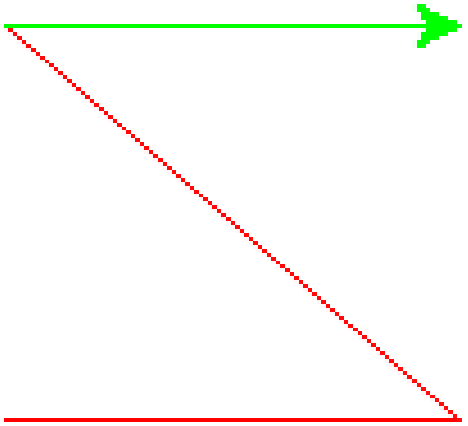

```
from turtle import Turtle

ruga1 = Turtle()
ruga1.color("red")
ruga1.forward(100)

ruga2 = Turtle()
ruga2.color("green")
ruga2.left(45)
ruga2.forward(100)

ruga3 = Turtle()
ruga3.color("yellow")
ruga3.right(45)
ruga3.forward(100)

ruga4 = Turtle()
ruga4.color("blue")
ruga4.left(180)
ruga4.forward(100)
```

```
ruga1.clear()   # cancella la linea
ruga2.reset()   # cancella e ripristina
```



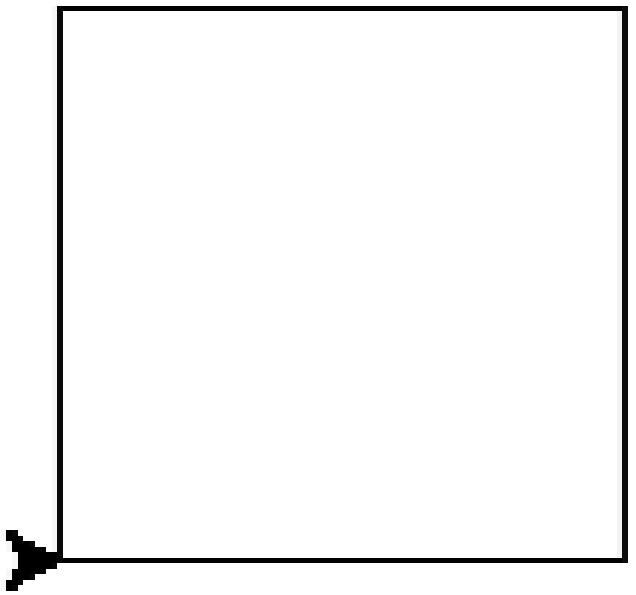


```
t.color("red")
t.forward(100)
t.goto(0,100)
t.color("green")
t.forward(100)
```



```
t.color("red")
t.forward(100)
t.up()
t.goto(0,100)
t.down()
t.color("green")
t.forward(100)
```

Disegnare un quadrato



```
t = Turtle()
lato=100
for i in range(4):
    t.forward(lato)
    t.left(90)
```

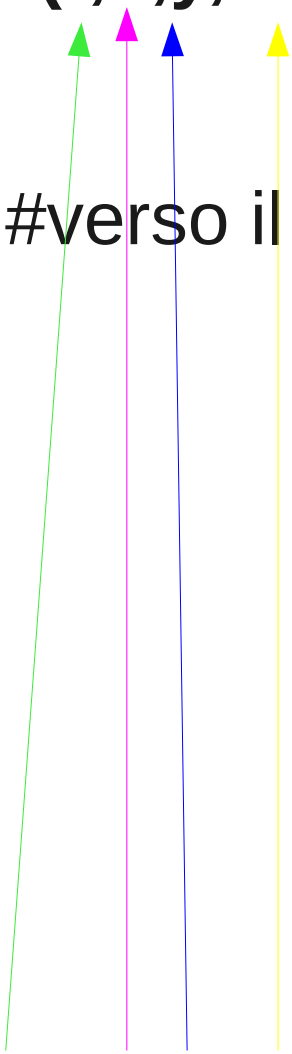
Recicliamo!


```
def saluta(n,c):
    print('Ciao',n,c)
    return

saluta('Mario','Rossi')
saluta('Pino','Bianchi')
```

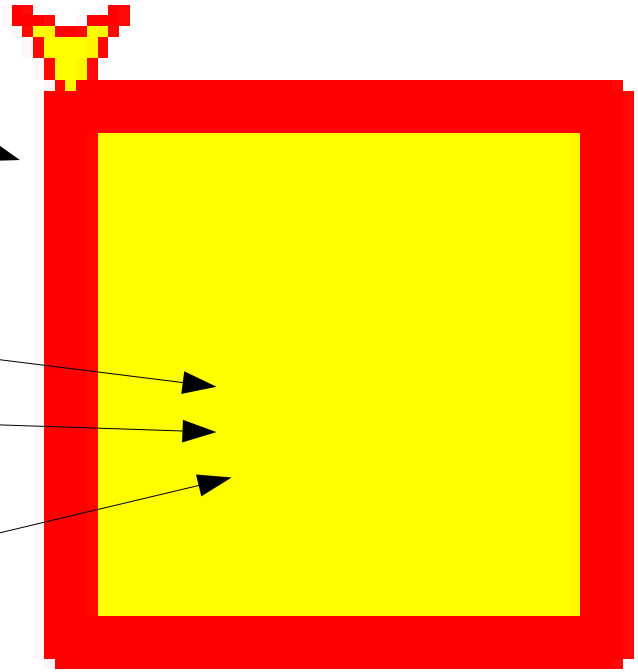
def

da non dimenticare!

```
from turtle import Turtle

def disegnaQuadrato(t,x,y,lato):
    t.up();
    t.goto(x,y);
    t.setheading(270); #verso il basso/sud
    t.down();
    for i in range(4):
        t.forward(lato)
        t.left(90)


# creo l'oggetto
ruga = Turtle()


disegnaQuadrato(ruga,1,1,50)
```



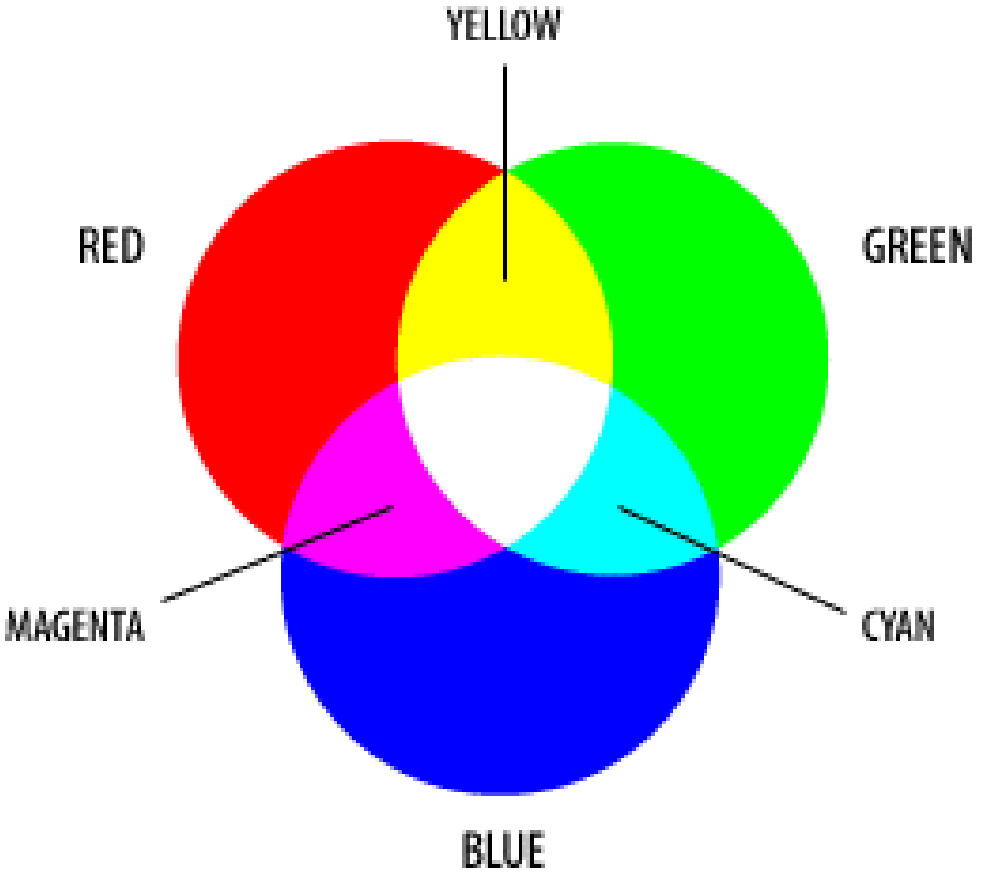
```
def disegnaQuadrato(t, x, y, lato, colore, coloreBordo):
    t.up()
    t.goto(x,y)
    t.pencolor(coloreBordo)
    t.setheading(270)
    t.down()
    t.width(5)
    t.fillcolor(colore)
    t.begin_fill()
    for i in range(4):
        t.forward(lato)
        t.left(90)
    t.end_fill()
```

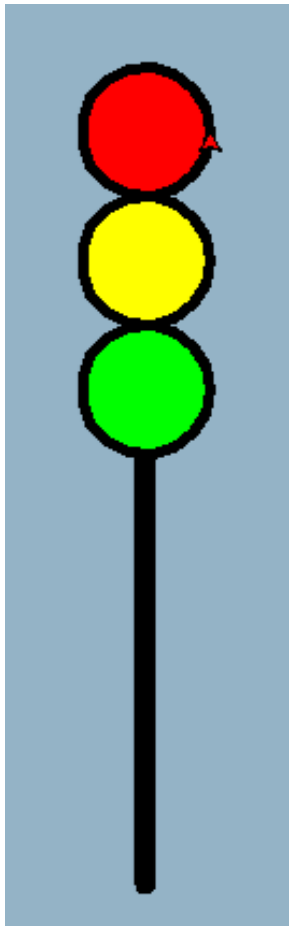


```
disegnaQuadrato(ruga, 1, 1, 50, "yellow", "red")
```

I colori

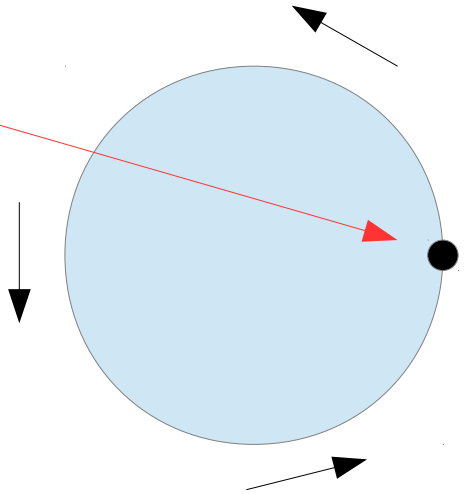
ruga.pencolor(“#FF0000”)



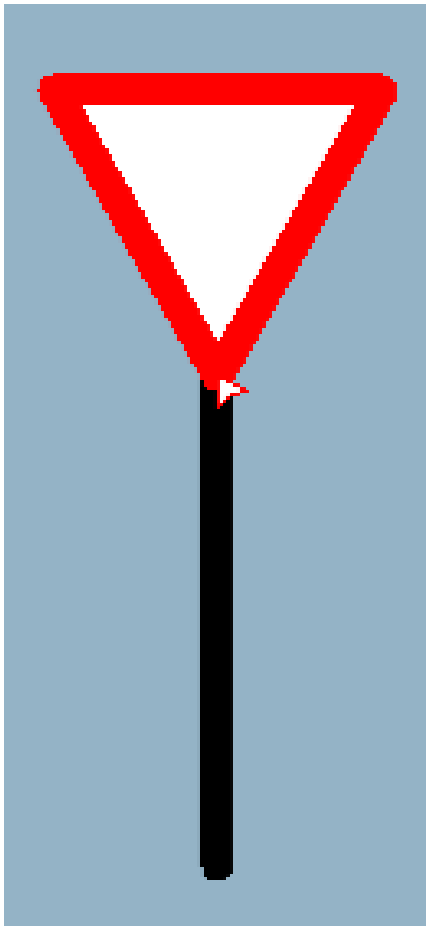


Lavoriamo sui cerchi

```
def disegnaCerchio(t, raggio, x, y, colore, coloreBordo):
    t.up()
    t.goto(x,y)
    t.fillcolor(colore)
    t.pencolor(coloreBordo)
    t.begin_fill()
    t.down()
    t.circle(raggio)
    t.end_fill()
```



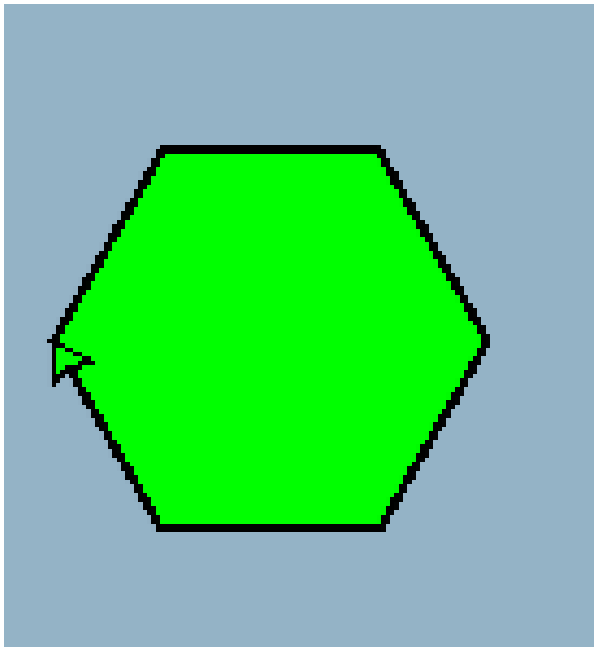
```
ruga.up()
ruga.goto(0,-200)
ruga.down()
ruga.width(10)
ruga.left(90)
ruga.forward(200)
ruga.width(5)
raggio=30
disegnaCerchio(ruga, raggio, ruga.xcor()+raggio, ruga.ycor()+raggio, "green", "black")
disegnaCerchio(ruga, raggio, ruga.xcor(), ruga.ycor()+raggio*2, "yellow", "black")
disegnaCerchio(ruga, raggio, ruga.xcor(), ruga.ycor()+raggio*2, "red", "black")
```



Lavoriamo sui triangoli

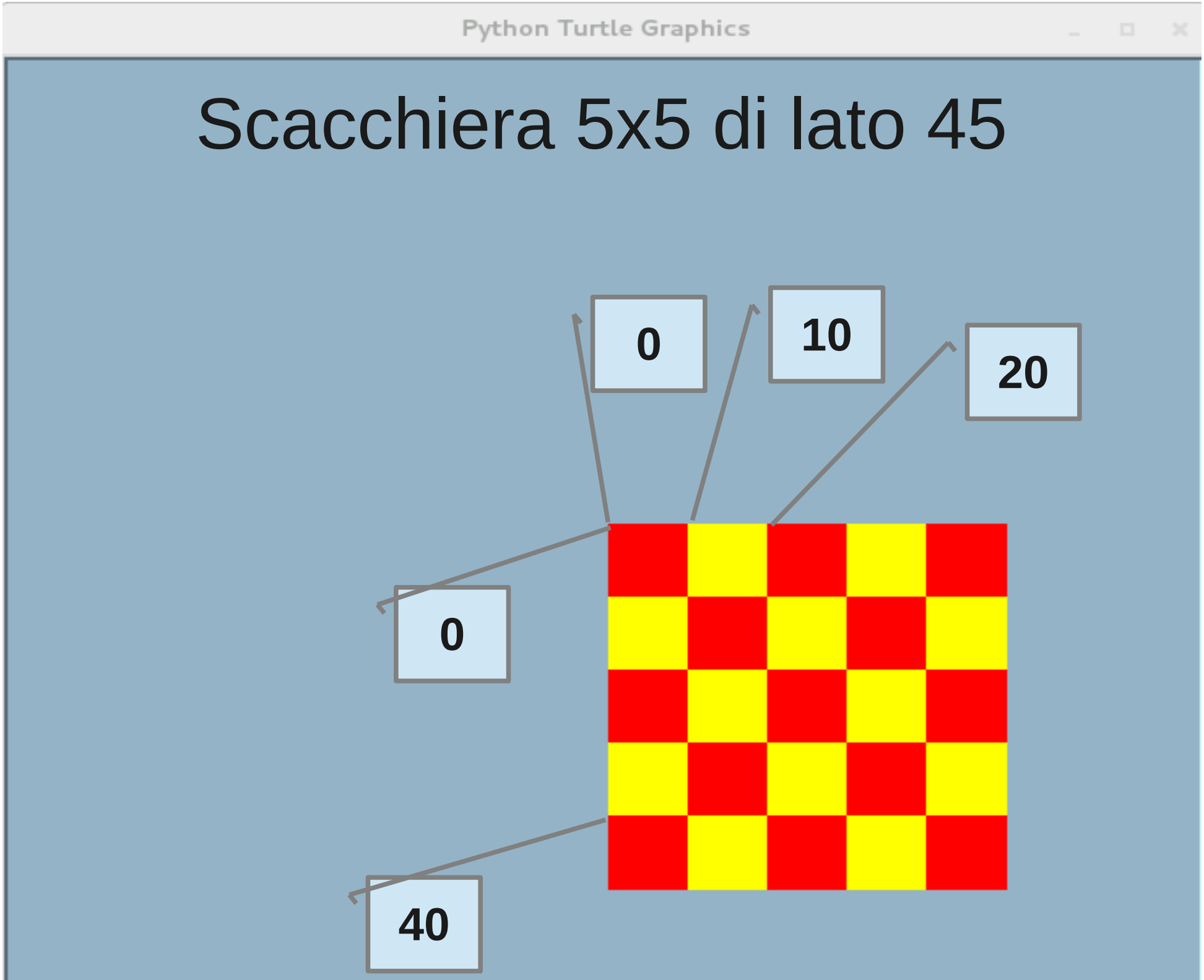
```
def disegnaTriangolo(t, x, y, lato, colore, coloreBordo):
    t.up()
    t.goto(x,y)
    t.pencolor(coloreBordo)
    t.fillcolor(colore)
    t.begin_fill()
    t.down()
    for i in range(3):
        t.forward(lato)
        t.right(360/3)
    t.end_fill()
```

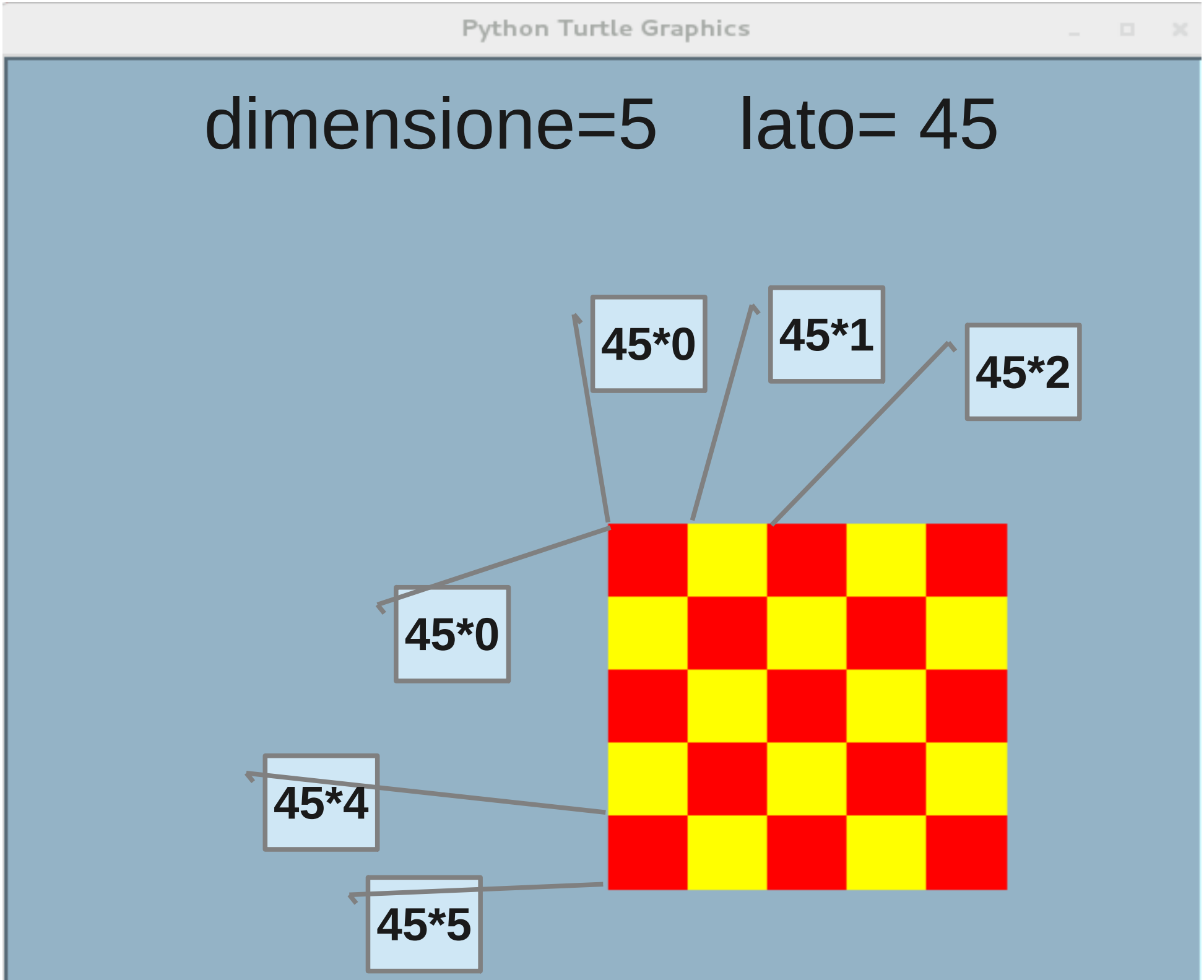
```
ruga.up()
ruga.goto(100,-200)
ruga.down()
ruga.width(10)
ruga.forward(150)
ruga.width(10)
ruga.left(30)
lato=100
disegnaTriangolo(ruga, ruga.xcor(), ruga.ycor(), lato, "white", "red")
```



Poligoni

```
def disegnaPoligono(t, x, y, numeroVertici, lato, colore, coloreBordo):
    t.up()
    t.goto(x,y)
    t.pencolor(coloreBordo)
    t.fillcolor(colore)
    t.begin_fill()
    t.down()
    for i in range(numeroVertici):
        t.backward(lato)
        t.left(360.0/numeroVertici)   # float
    t.end_fill()

disegnaPoligono(ruga, -200, 0, nvertici, lato, "green", "black")
```

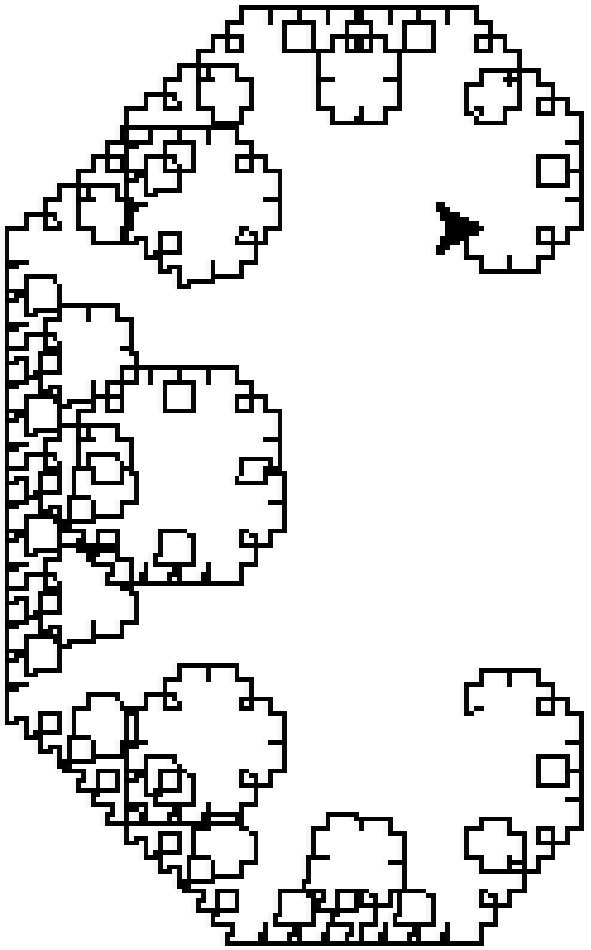

Python Turtle Graphics
Scacchiera 5x5 di lato 45
0
10
20
0
40


Python Turtle Graphics
dimensione=5 lato= 45
45*0
45*1
45*2
45*0
45*4
45*5

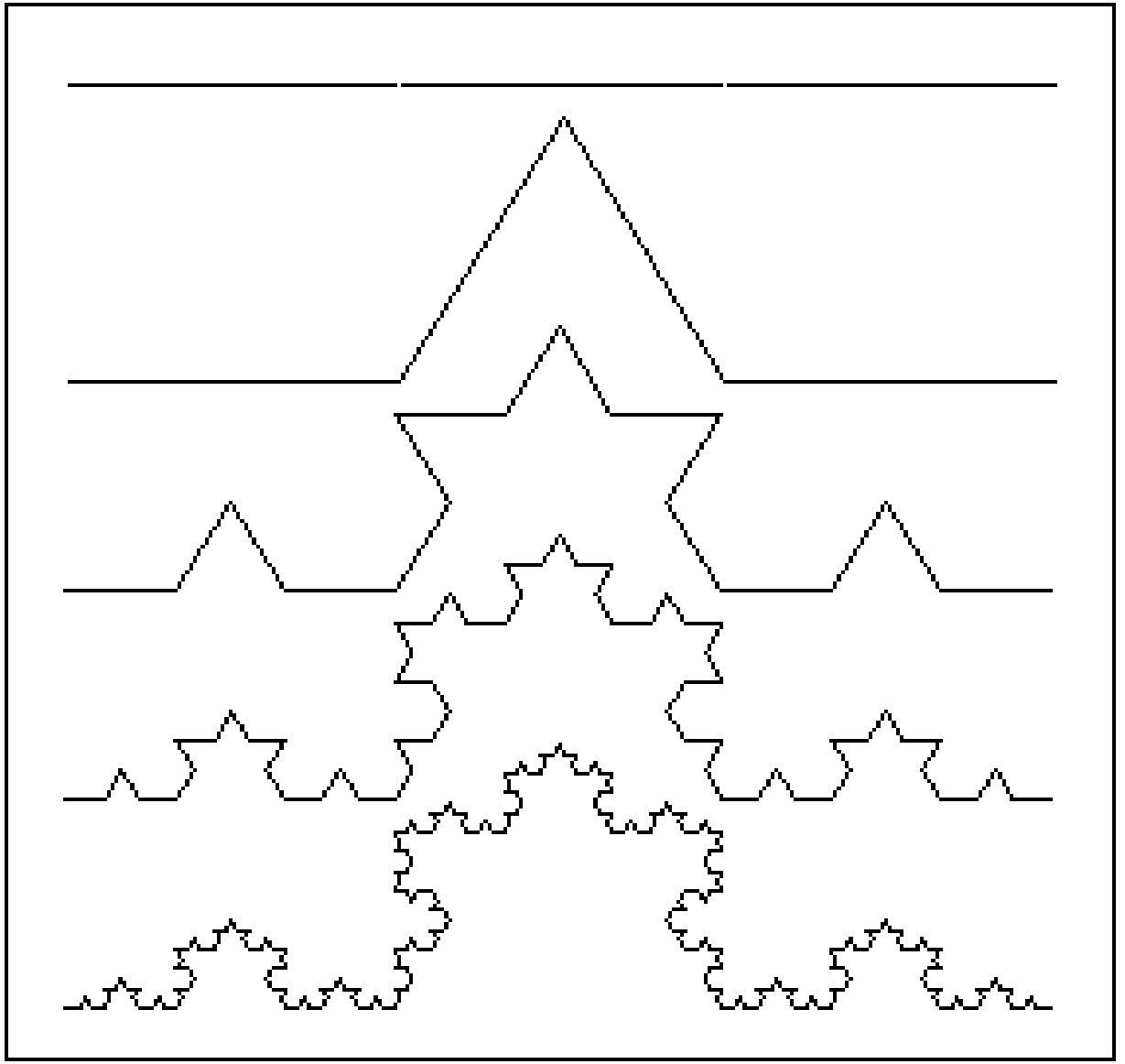
```
from turtle import *

# creo una tartaruga
ruga = Turtle()

#imposto i colori
ruga.screen.bgcolor("#94B3C6")

def disegnaQuadrato(t,x,y,lato,colore):
    t.up()
    t.goto(x,y)
    ruga.color(colore)  # bordo
    t.fillcolor(colore)     # riempimento
    t.begin_fill()
    t.down()
    for i in range(4):  # 4 lati
        t.forward(lato)
        t.right(90)
    t.end_fill()
```

```
# muovo la tartaruga
lato=int(input("Inserire la lunghezza del lato: "))
dimensione=int(input("Inserire la dimensione: "))
max=dimensione*lato

# 10 righe di lato 10: x va da da 0 alla posizione corrente
# al secondo meno 1 99
# incremento di 10: 0, 10, 20, 30, 40, 50, 60, 70, 80, 90
# range(1, 6, 2) = 1, 3, 5
for posizione_x in range(0,max,lato):
    for posizione_y in range(0,-max,-lato):
        if ((posizione_x+posizione_y)/lato) %2 == 0:
            disegnaQuadrato(ruga, posizione_x, posizione_y, lato,
"red") #in pixel
        else:
            disegnaQuadrato(ruga, posizione_x, posizione_y, lato,
"yellow")      #in pixel
ruga.hideturtle()
```

Frattali e ricorsione



Curva a fiocco di neve



Curva di livello N

Dati due punti (x1, y1) e (x2, y2) e un livello:

se livello = 0

- si uniscono i punti con un segmento di retta

sltrimenti

- si trova un punto (xm,ym)

xm = (x1 + x2 + y1 - y2) // 2

ym = (x2 + y1 + y2 - x1) // 2

- si richiama la costruzione della curva due volte con:

(x1,y1) e (xm,ym)

(xm,ym) e (x2,y2)

```
from turtle import Turtle

def disegnaLinea(x1, y1, x2, y2):
    t.up()
    t.goto(x1, y1)
    t.down()
    t.goto(x2, y2)

def disegnaCurva(t, x1, y1, x2, y2, livello):
    if livello == 0:
        disegnaLinea(x1, y1, x2, y2)
    else:
        xm = (x1 + x2 + y1 - y2) // 2
        ym = (x2 + y1 + y2 - x1) // 2
        disegnaCurva(t, x1, y1, xm, ym, livello-1)
        disegnaCurva(t, xm, ym, x2, y2, livello-1)

livello = int(input("Inserisci il livello (0 o maggiore): "))
t = Turtle()
disegnaCurva(t, 50, -50, 50, 50, livello)
```