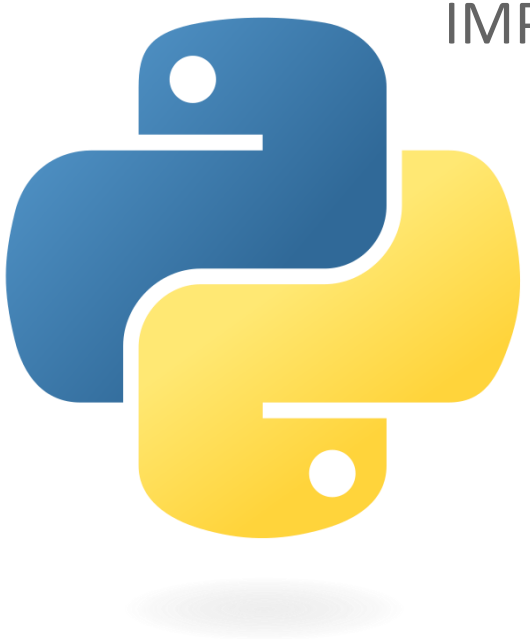


IMPARIAMO A PROGRAMMARE IN python™

LEZIONE 3

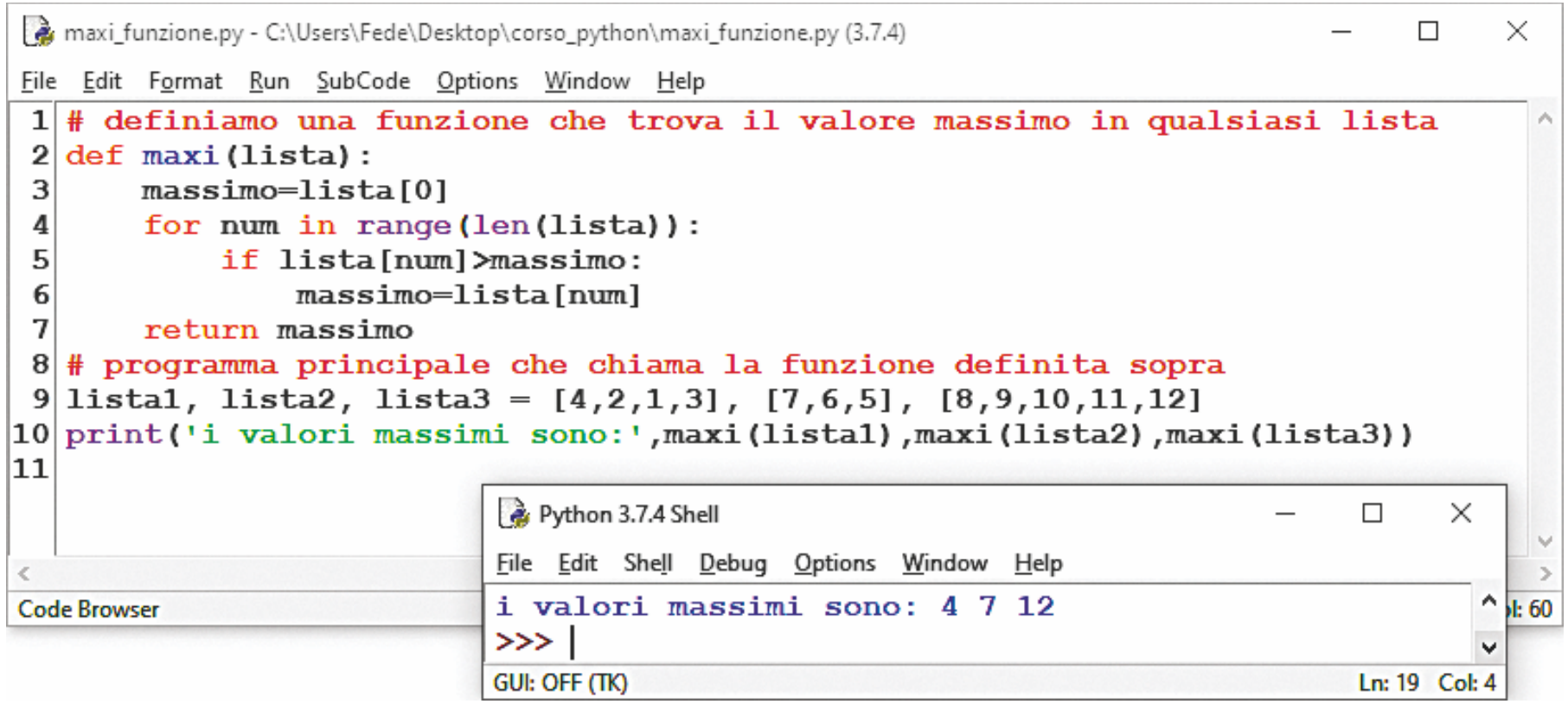
Capitolo 3 – Le funzioni

Indice

3.1 L'uso delle funzioni

Indice

Una funzione è un gruppo di istruzioni che si può eseguire ogni volta che serve, scrivendolo una volta sola.



3.1 L’uso delle funzioni

Indice

In Python una funzione è composta da un’intestazione, seguita da un blocco di istruzioni indentate.

def nomeFunzione(*argomento1,argomento2,...,argomentoN):*

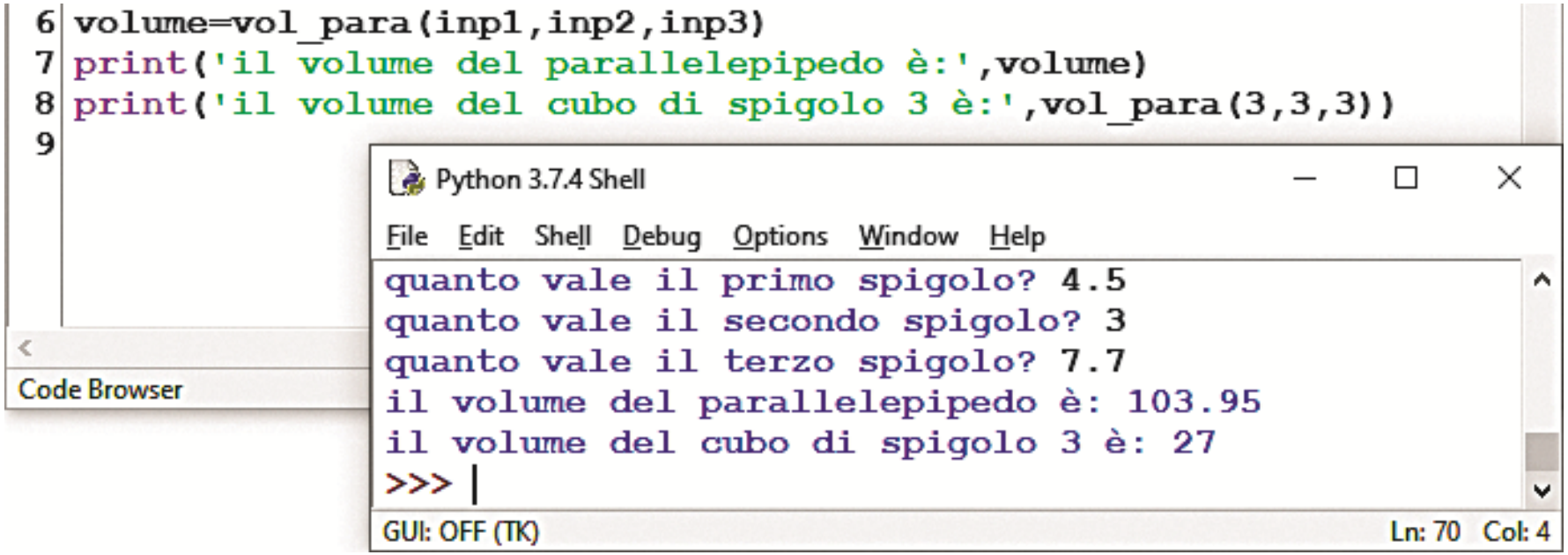
Di norma, una funzione riceve come input alcuni dati dal programma principale e li elabora calcolando un valore; poi, alla fine della sua esecuzione, restituisce questo dato di output al programma principale che l’ha chiamata attraverso un’operazione di `return`.

```
def maxi(lista):
    massimo=lista[0]
    for num in range(len(lista)):
        if lista[num]>massimo:
            massimo=lista[num]
    return massimo
```

3.1 L'uso delle funzioni

Indice

Una funzione si chiama passandole dei valori per gli argomenti. Essa li elabora e restituisce al programma principale uno o più valori di ritorno.

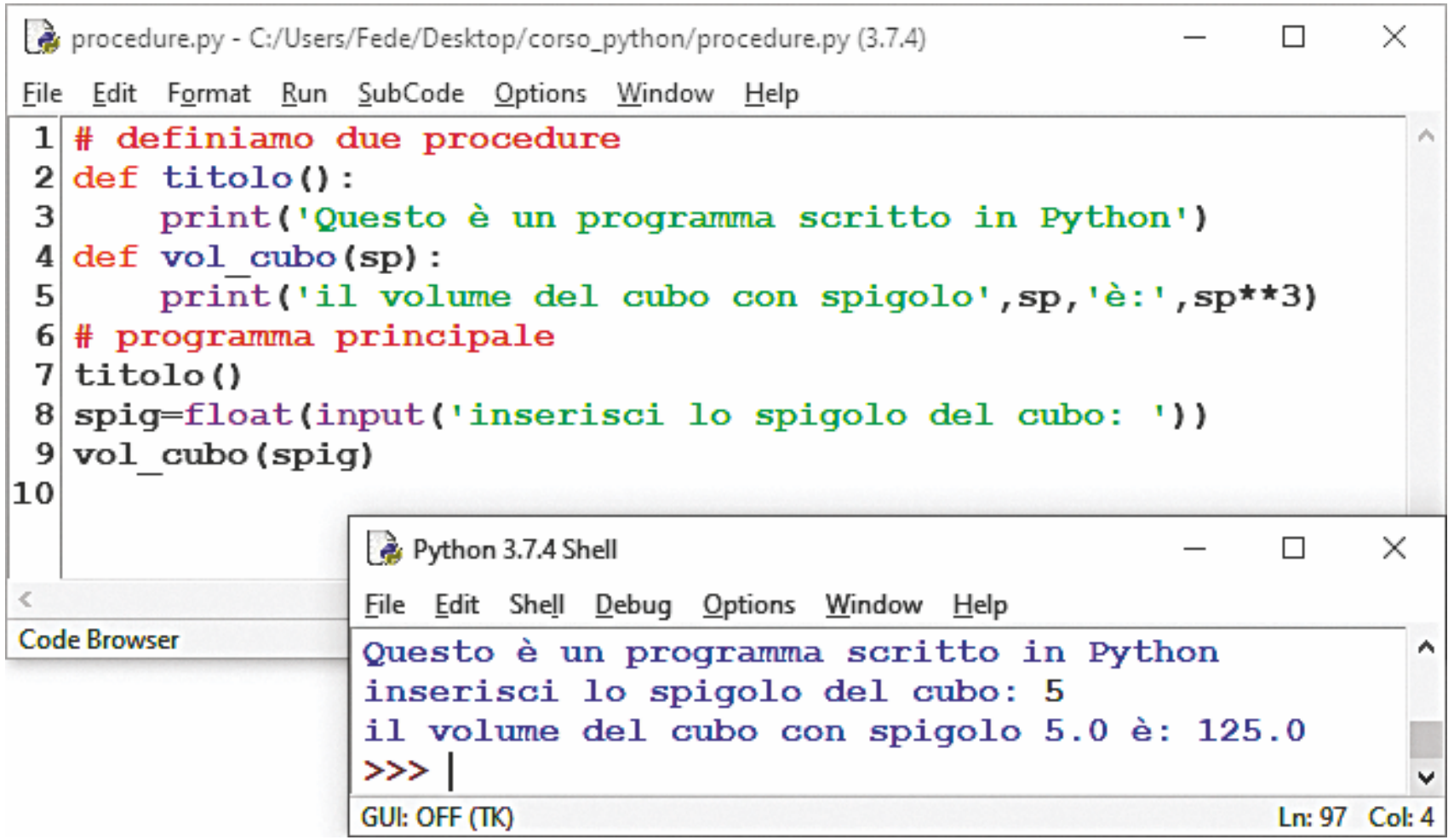


Nel caso in cui i valori siano più di uno, vengono restituiti sotto forma di tupla.

3.2 Funzioni particolari

Indice

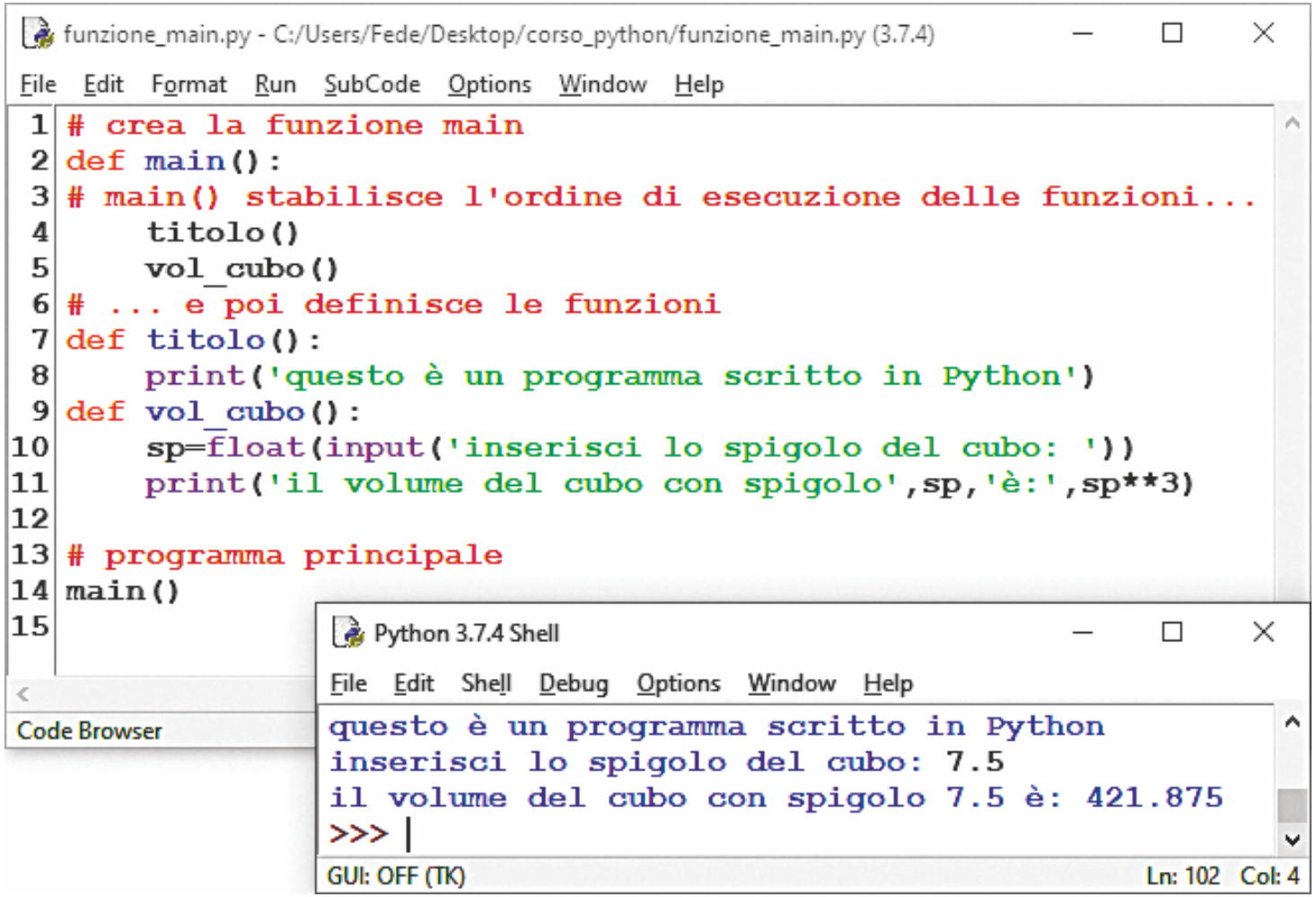
Una procedura è una funzione che non produce valori.



3.2 Funzioni particolari

Indice

In Python spesso si crea una funzione principale `main()` che chiama le altre funzioni del programma.

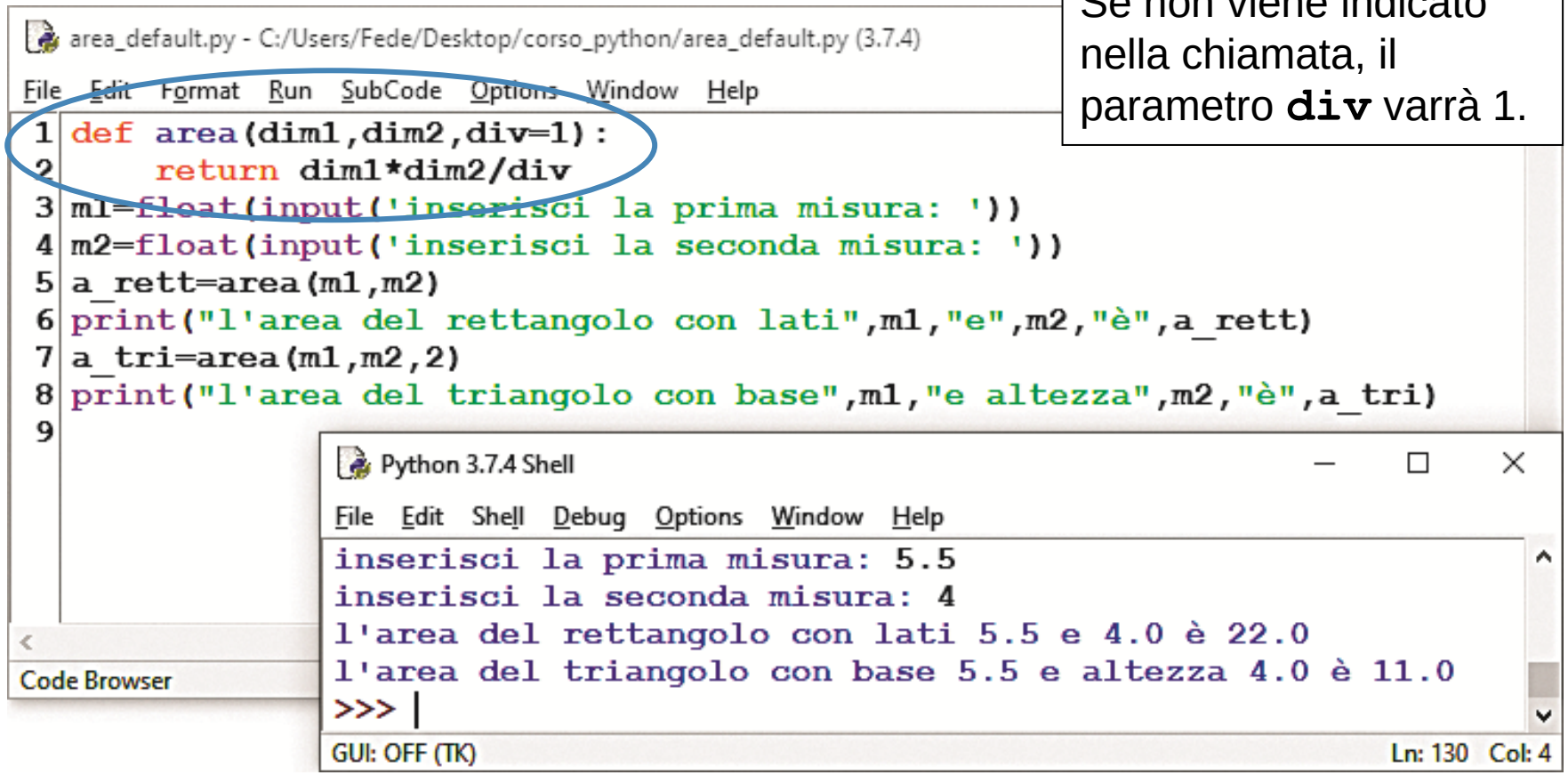


3.2 Funzioni particolari

Indice

Nell'intestazione di una funzione si possono indicare valori di default per i parametri.

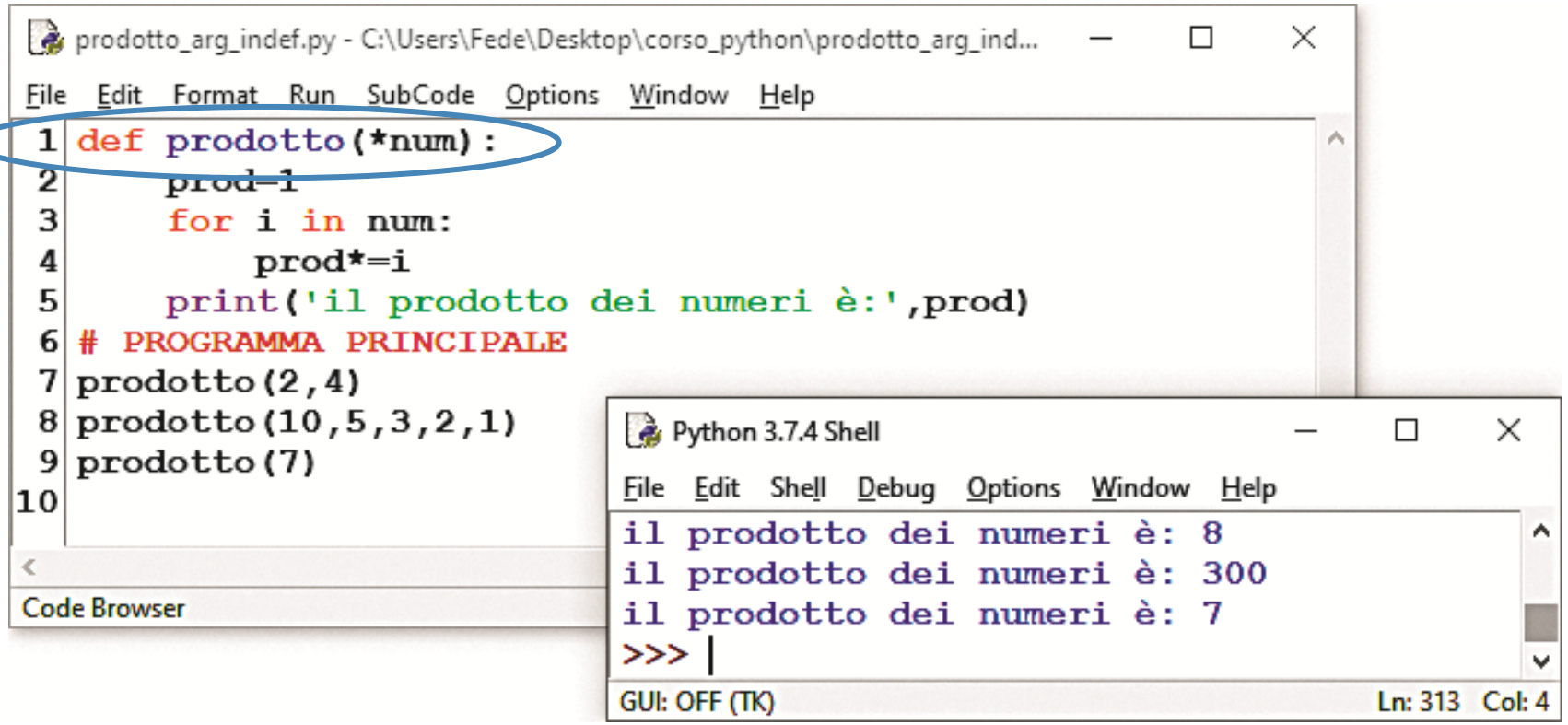
Se non viene indicato nella chiamata, il parametro **`div`** varrà 1.



3.2 Funzioni particolari

Indice

Una funzione può ricevere un numero variabile di parametri.



3.3 Gli ambiti di visibilità

Indice

Ogni variabile ha un ambito di visibilità (scope) all'interno del quale è visibile o utilizzabile.

VARIABILI LOCALI	VARIABILI GLOBALI
Dichiarate nelle funzioni. Sono viste nella funzione e nelle eventuali funzioni annidate di essa	Dichiarate nel programma principale. Visibili nell'intero file.

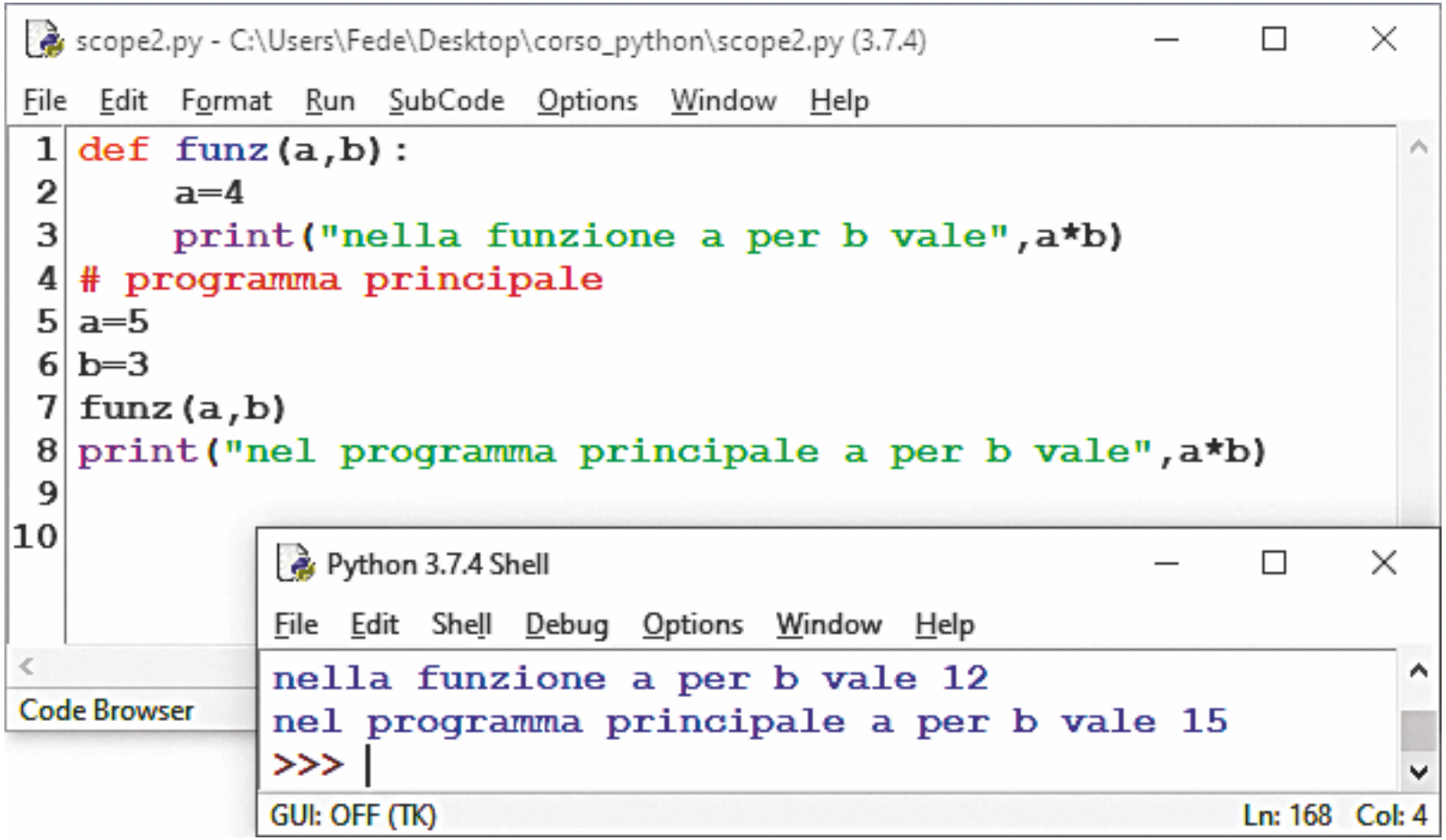
```
def funz():
    print("nella funzione a per b vale",a*b)
# programma principale
a=5
b=3
funz()
```

La funzione può utilizzare le variabili globali.

3.3 Gli ambiti di visibilità

Indice

Nel programma principale non sono visibili le variabili della funzione, che sono locali.



3.3 Gli ambiti di visibilità

Indice

Quando una funzione all'interno del suo blocco di istruzioni definisce un'altra funzione, questa viene detta funzione annidata e non è visibile dal programma principale.

```
def esterna():
    a=5
    def interna():
        print("per la funzione interna a vale",a)
        b=3
    interna()
    #print("per la funzione esterna b vale",b)

esterna()
interna()
```

Questa istruzione causa un'eccezione, perché la funzione annidata non è riconosciuta dal programma principale.

3.4 I moduli

Indice

I moduli sono raccolte di funzioni pronte per l'uso.

Per utilizzare le funzioni contenute in un modulo bisogna importarlo con l'istruzione `import` nomeModulo.

Per chiamare una funzione appartenente a un modulo si usa nomeModulo.nomeFunzione(argomenti).

```
# proviamo a usare il modulo math
import math
# attribuiamo il risultato della funzione a una variabile
var=math.sqrt(5.5)
```

Da un modulo si possono importare singole funzioni.
In questo caso la funzione viene richiamata senza il nome del modulo.

```
from math import hypot

    hypot(cateto1,cateto2)
```

3.4 I moduli

Indice

Per abbreviare il codice si possono usare degli alias per i nomi delle funzioni e dei moduli.

```
# definiamo un alias
rr=random.randint
print('numero a caso tra 1 e 10:',rr(1,10))
```

`Import random as r`

Definisce **`r`** come alias di tutto il modulo **`random`**.

`From random import randint as r`

Definisce **`r`** come alias della funzione **`randint`** appartenente al modulo **`random`**.

3.4 I moduli

Indice

Si possono creare moduli personalizzati da salvare nella cartella che contiene il programma o in una cartella di sistema di Python.

```
# NUOVO MODULO CREATO DAL PROGRAMMATORE
# prima funzione
def inserisci(stringa):
    print(stringa)
    voto = float(input('inserisci il dato: '))
    return voto
# seconda funzione
def media(o,s):
    return(o+s)/2
```

Il modulo potrà essere importato normalmente nel programma principale.

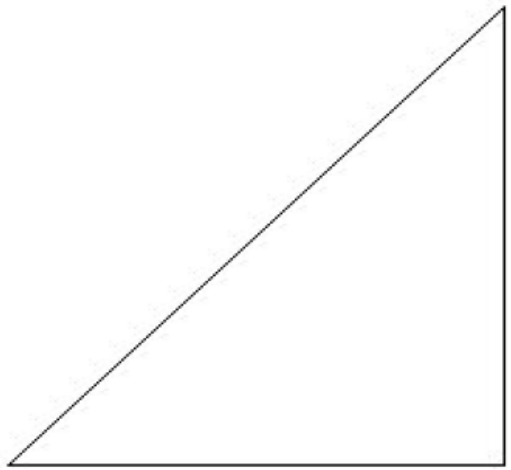
```
print('\t\tTEST! La media tra 4.5 e 7.5 è',mio_modulo.media(4.5,7.5))
```

3.5 Disegnare con `turtle`

Indice

Il modulo `turtle` mette a disposizione del programmatore funzioni per disegnare.

```
import turtle as t
t.showturtle()
t.forward(200)
t.left(90)
t.forward(200)
t.left(135)
import math
t.forward(math.hypot(200,200))
t.hideturtle()
```

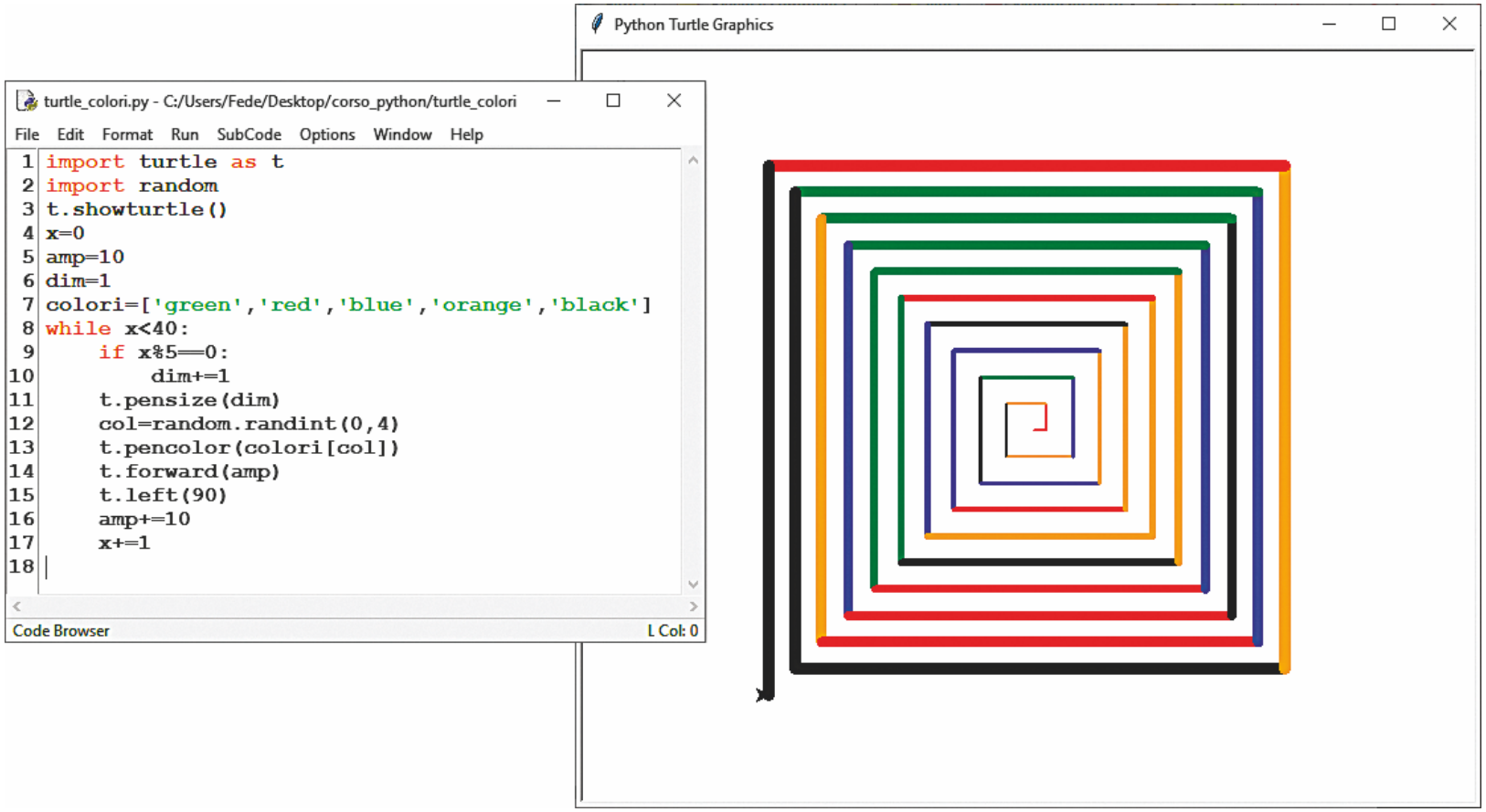


La «tartaruga» rappresenta l'oggetto che scrive

3.5 Disegnare con `turtle`

Indice

La «tartaruga» ha una penna che può scrivere in vari colori.



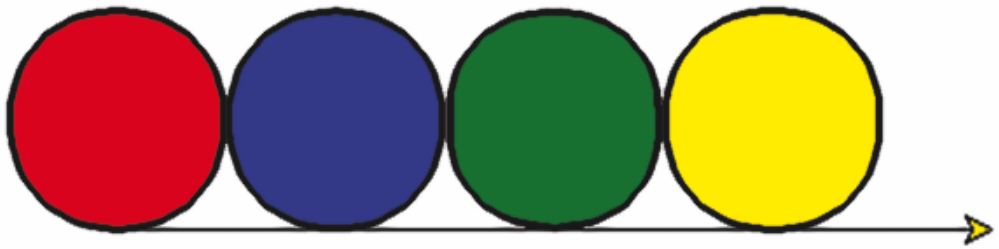
Le funzioni **`pensize()`** e **`pencolor()`** permettono di gestire lo spessore del tratto e il suo colore.

3.5 Disegnare con `turtle`

Indice

E' possibile anche riempire le forme disegnate.

```
import turtle as t
t.showturtle()
r=40
colori=['red','blue','green','yellow']
for col in colori:
    t.fillcolor(col)
    t.begin_fill()
    t.circle(r)
    t.end_fill()
    t.forward(r*2)
```



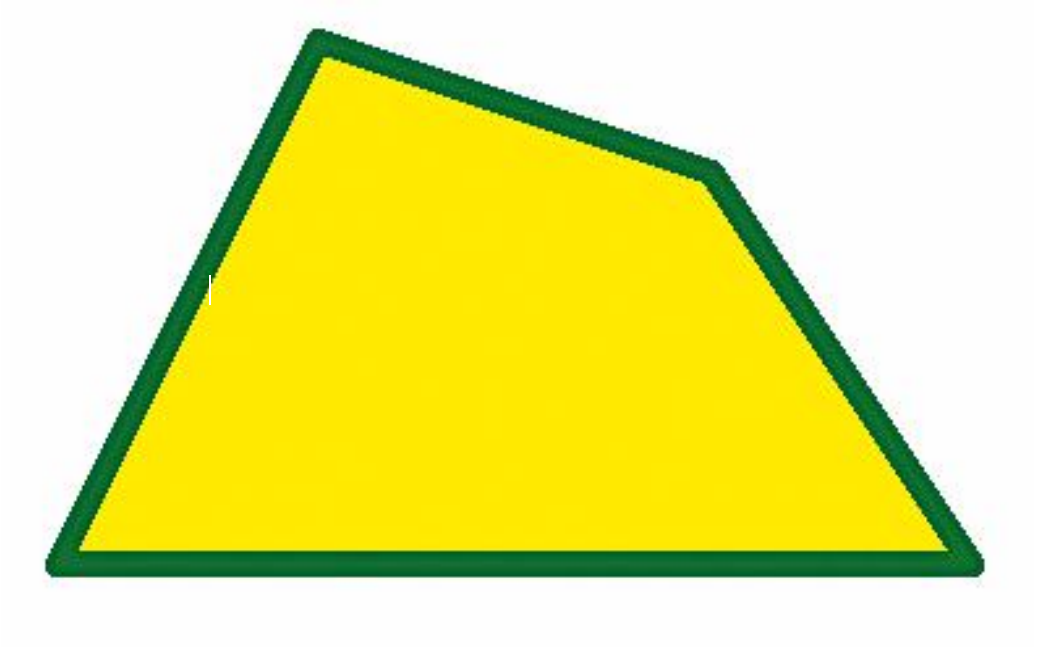
La funzione **`fillcolor()`** definisce il colore di riempimento,
`begin_fill()` specifica che la figura creata in seguito deve essere riempita.

3.5 Disegnare con `turtle`

Indice

La tartaruga disegna su di un piano cartesiano la cui origine è nel centro della finestra grafica.

```
import turtle as t
t.showturtle()
t.pencolor('green')
t.pensize(10)
t.fillcolor('yellow')
t.begin_fill()
t.penup()
t.goto(100,100)
t.pendown()
t.goto(-50,150)
t.goto(-150,-50)
t.goto(200,-50)
t.goto(100,100)
t.end_fill()
t.hideturtle()
```



La funzione **`goto()`** permette di portare la tartaruga nella posizione desiderata.