

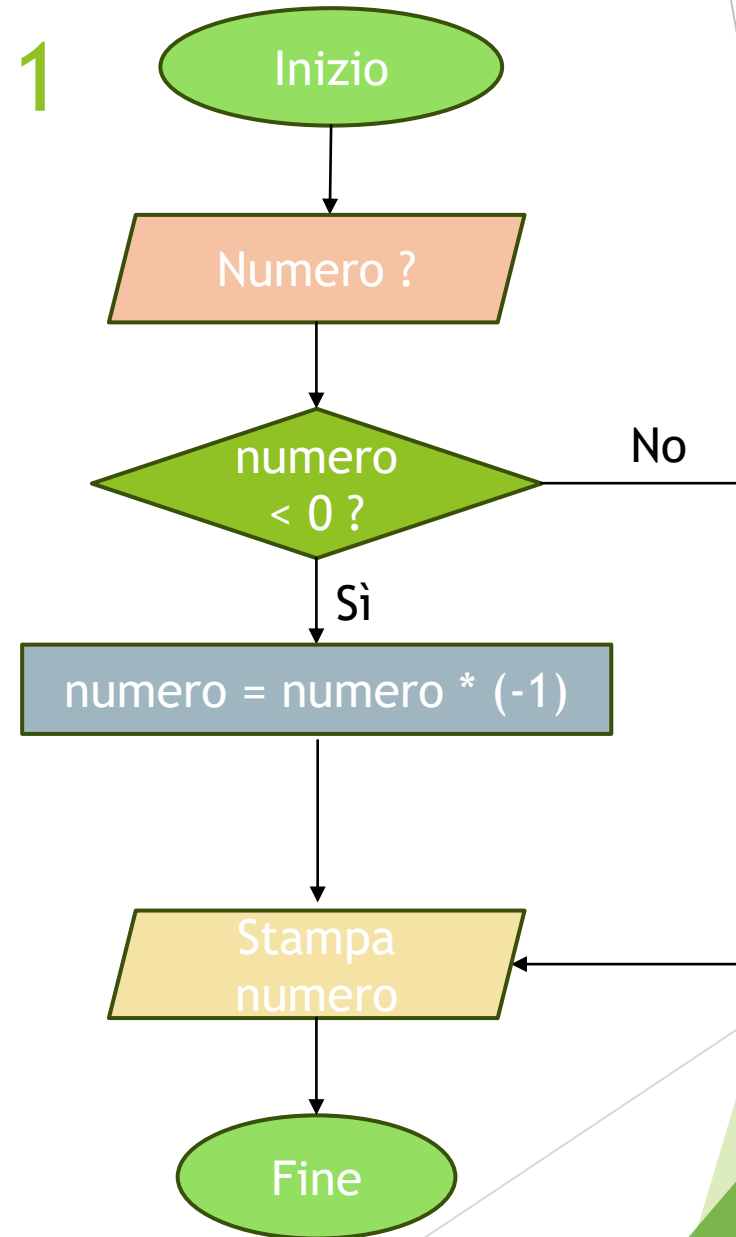
PYTHON BOOTCAMP

Lezione 2



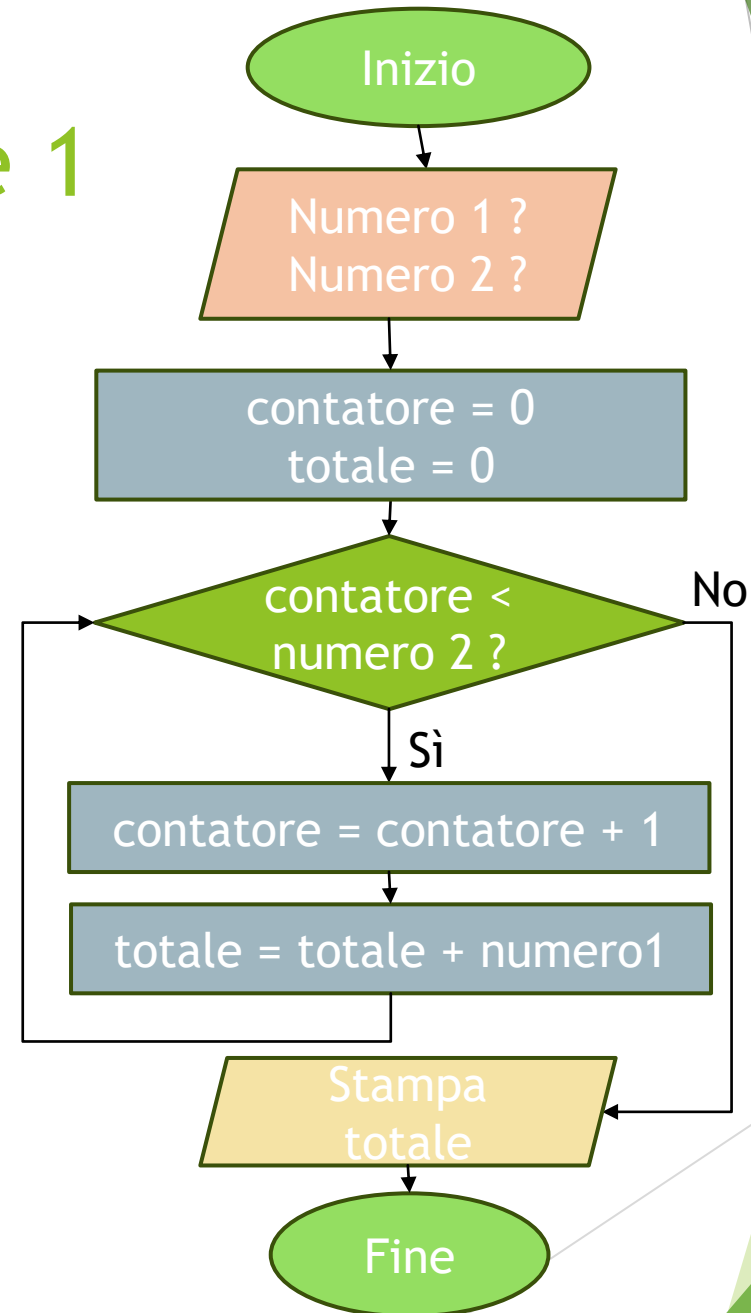
Soluzioni esercizi lezione 1

► Esercizio valore assoluto



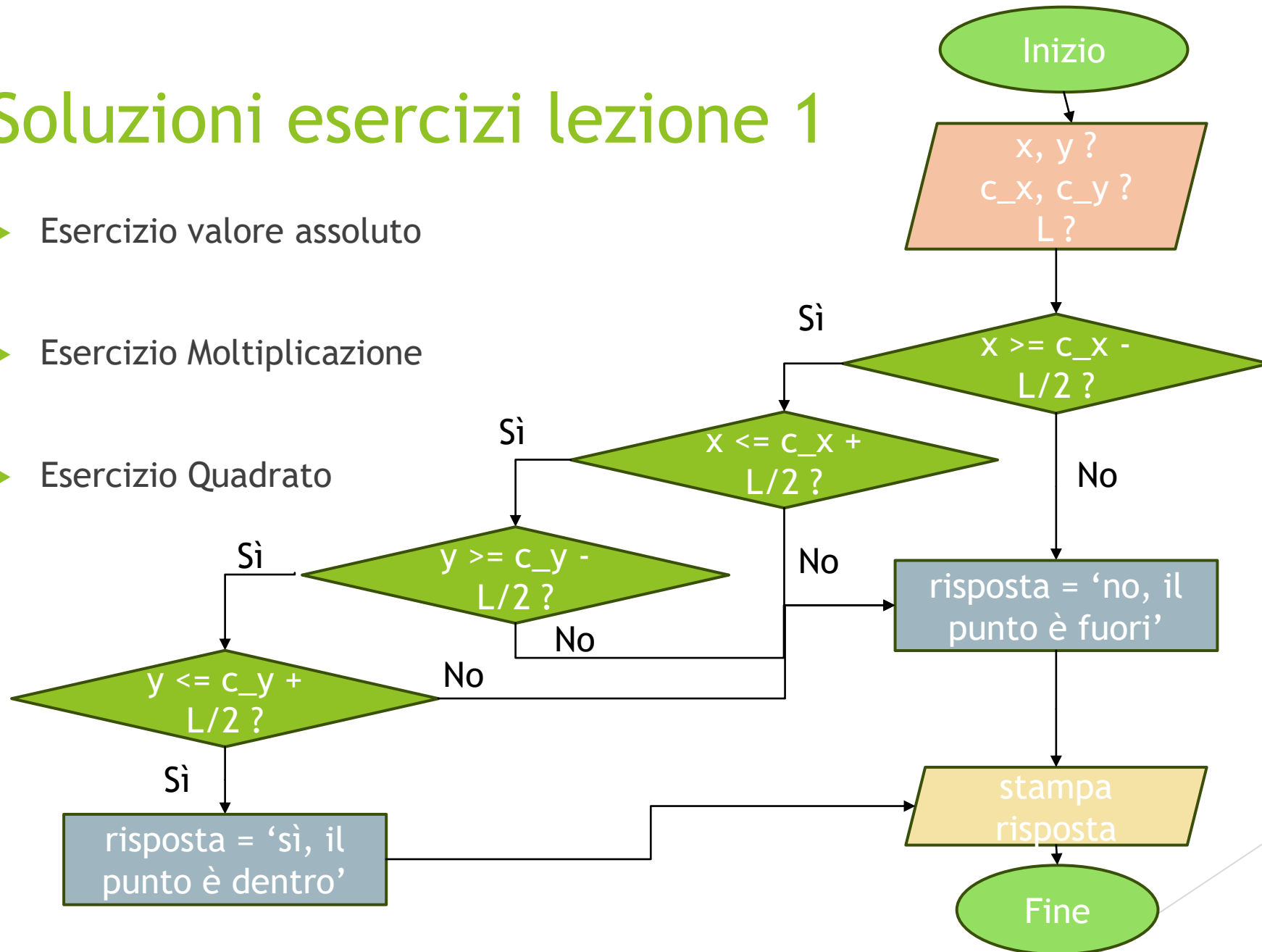
Soluzioni esercizi lezione 1

- Esercizio valore assoluto
- Esercizio Moltiplicazione



Soluzioni esercizi lezione 1

- Esercizio valore assoluto
- Esercizio Moltiplicazione
- Esercizio Quadrato

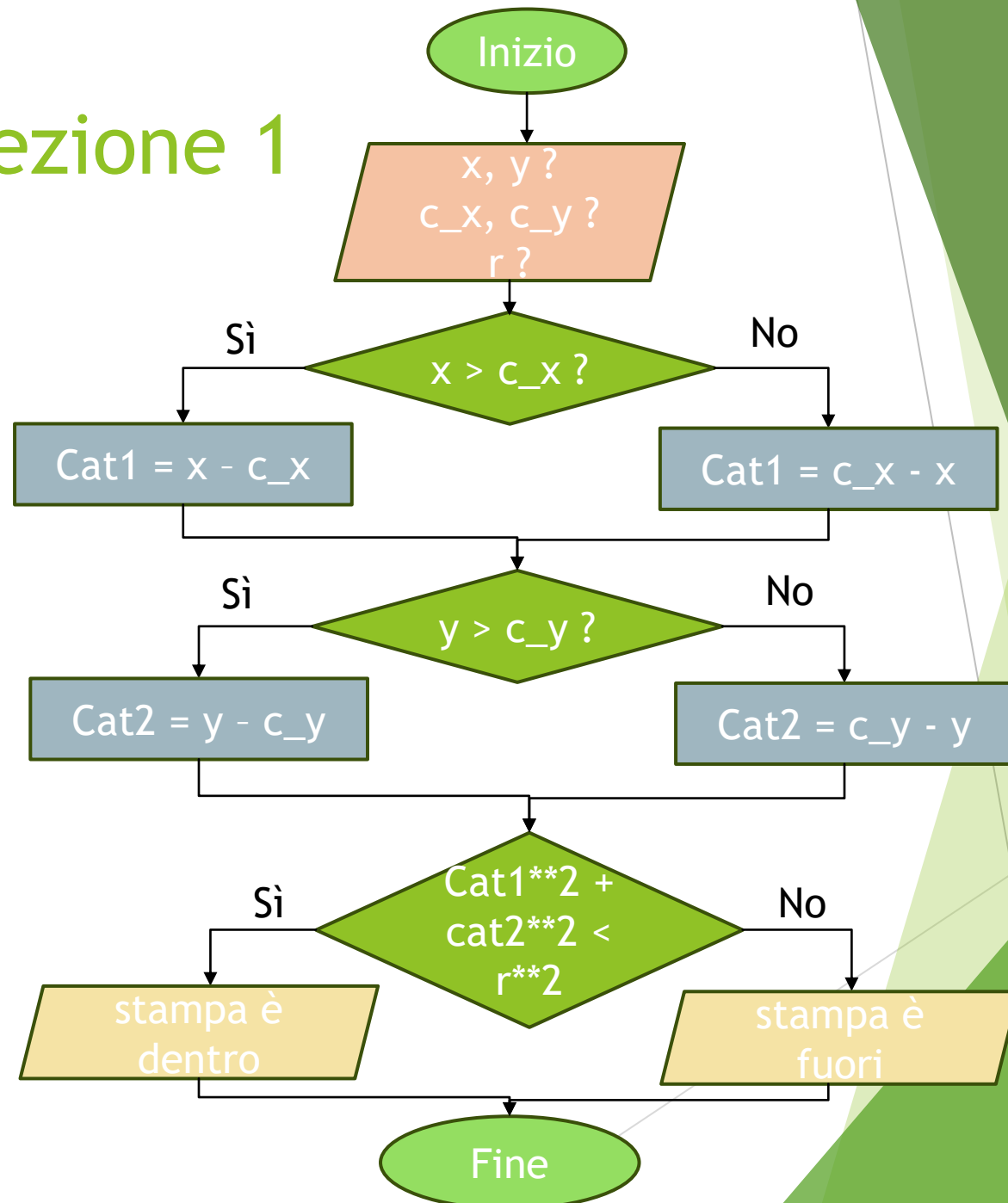


Soluzioni esercizi lezione 1

- ▶ Esercizio valore assoluto
- ▶ Esercizio Moltiplicazione
- ▶ Esercizio Quadrato: C'è un modo di farlo più compatto -> uso gli and!

Soluzioni esercizi lezione 1

- Esercizio valore assoluto
- Esercizio Moltiplicazione
- Esercizio Quadrato
- Esercizio Cerchio



Soluzioni esercizi lezione 1

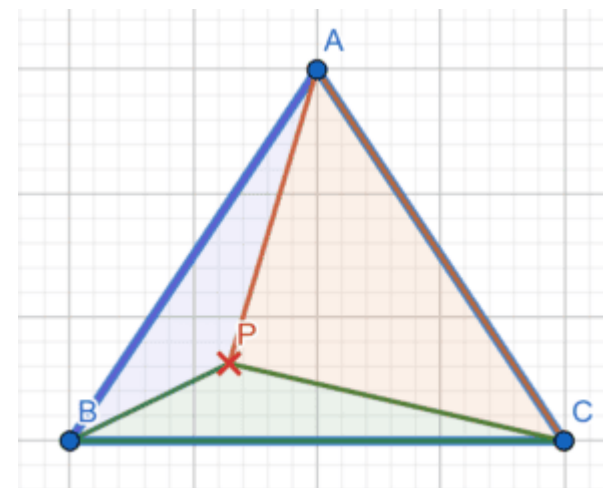
► Esercizio valore assoluto

► Esercizio Moltiplicazione

► Esercizio Quadrato

► Esercizio Cerchio

► Triangolo? Se A, B e C sono i vertici del triangolo e P il punto inserito, le aree dei triangoli APB, APC e BPC devono essere uguali all'area del triangolo ABC. Se questo è vero, il punto P è dentro al triangolo ABC

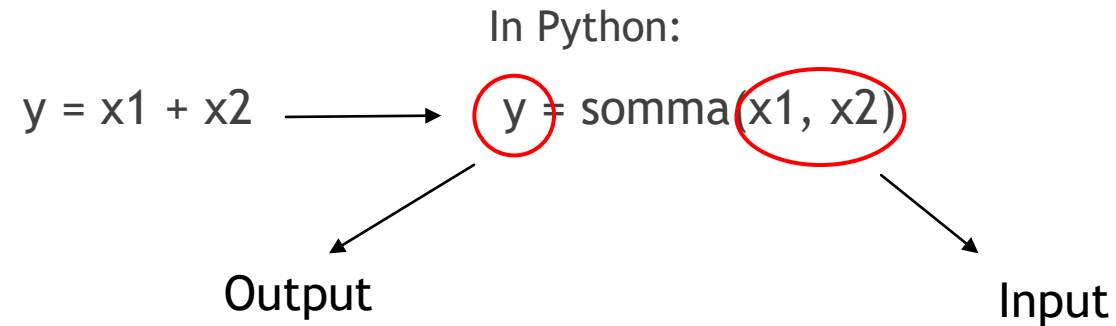


FUNZIONI



FUNZIONI

- ▶ Tramite le funzioni posso accorpare porzioni di codice che devo riutilizzare più volte
- ▶ Le funzioni hanno input e output, come nelle funzioni matematiche:



- ▶ Per definire una funzione devo usare una sintassi specifica

FUNZIONI

Inizio della funzione

funzione

Script

```
def somma(a, b):  
    risultato = a + b  
    return risultato  
  
x1 = 3  
x2 = 5  
  
risultato_somma = somma(x1, x2)  
print(risultato_somma)  
nuovo_risultato_somma = somma(4, 7)  
print(nuovo_risultato_somma )
```

Fine della funzione

Chiamata della funzione

Shell

```
8  
11  
>>>
```

BLOCCHI



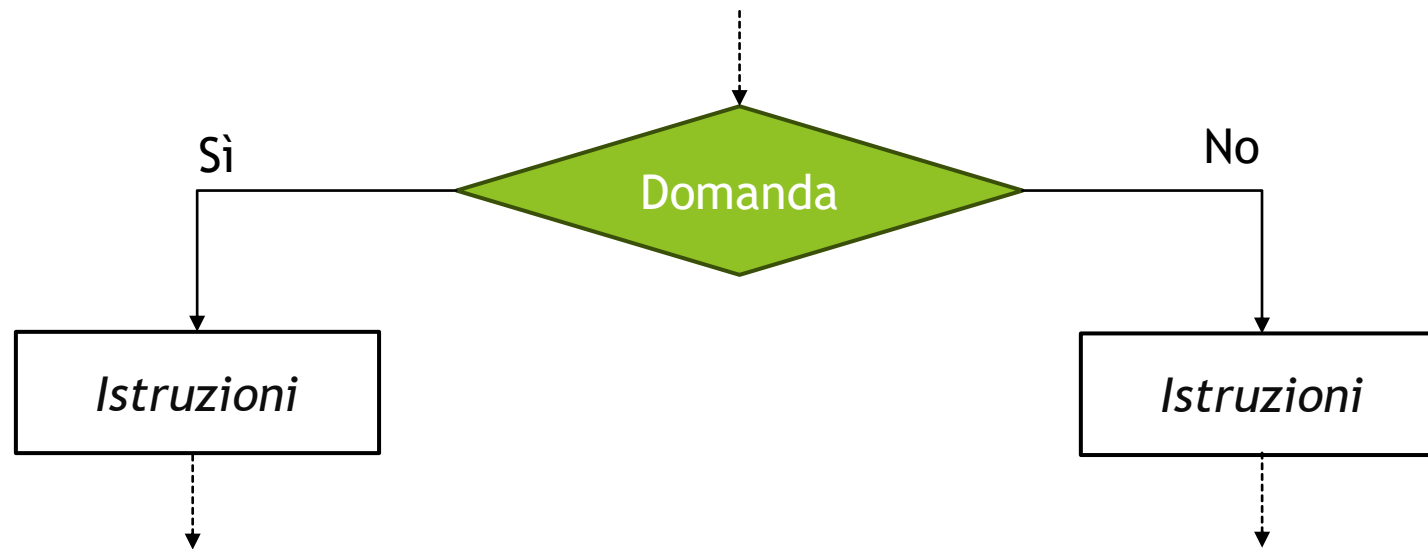
BLOCCHI

- Concetto importante in python: blocchi
- Ogni blocco è definito dal tasto *Tab*, tutte le porzioni di codice allineate allo stesso tab sono blocchi
- *Es. tutto ciò che è nel blocco della funzione deve iniziare alla stessa colonna*
- Una volta definita la funzione posso chiamarla tutte le volte che voglio e nell'ordine che voglio
- Le variabili definite nella funzioni sono utilizzabili solo nel blocco della funzione

Script

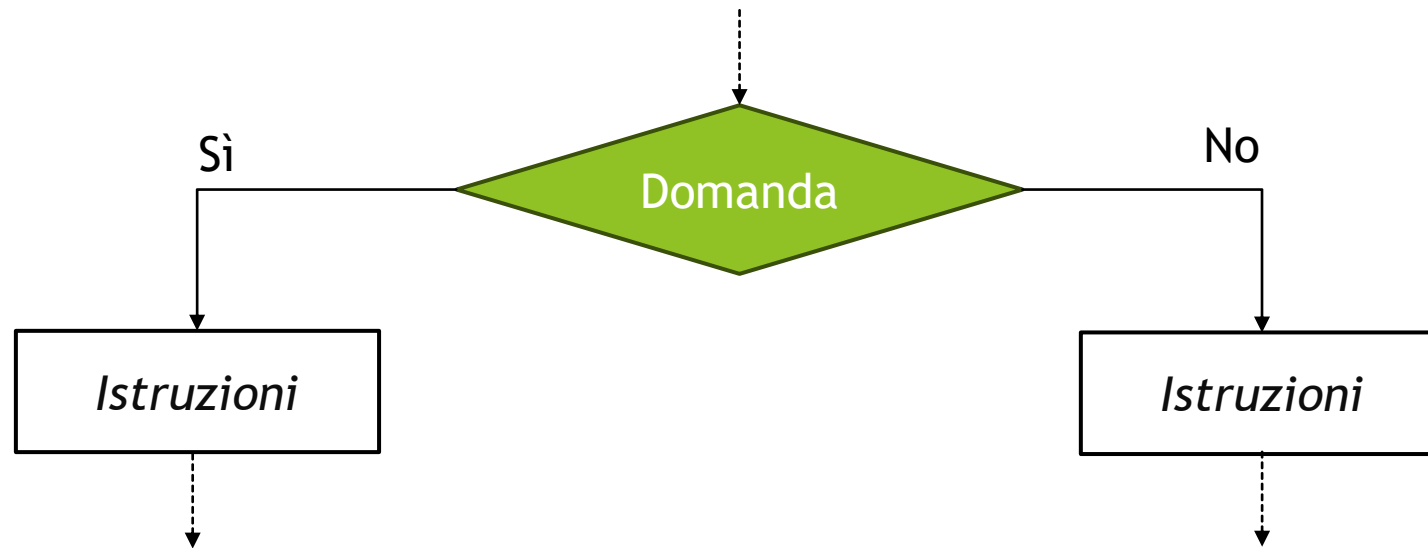
```
def somma(a, b):  
    risultato = a + b  
    return risultato  
  
def divisione(a, b):  
    risultato = a / b  
    return risultato  
  
risultato_divisione = divisione(30, 3)  
risultato_somma = somma(34, 73)
```

CONDIZIONI: if... else...



CONDIZIONI: if... else...

- ▶ Accedo a una porzione di codice solo se una condizione è verificata!
Se ... Allora ... Altrimenti...
- ▶ Devo tenere a mente tutte le possibili casistiche



CONDIZIONI: if... else...

- ▶ La sintassi è la seguente:
- ▶ Non è necessario mettere *else*! Se non avviene la condizione non faccio nulla
- ▶ Posso aggiungere anche più if di seguito con condizioni diverse, devono essere condizioni alternative!!!

Script

... codice ...
If *condizione*:

if
blocco

...

```
else:
```

```

        blocco
    else

```

...

CONDIZIONI

Shell

```
>>> anno_di_nascita = 2005  
>>> if anno_di_nascita < 2006:  
    print('Sei maggiorenne!')
```

Sei maggiorenne!

CONDIZIONI

Shell

```
>>> anno_di_nascita = 2005  
>>> if anno_di_nascita < 2006:  
    print('Sei maggiorenne!')
```

Sei maggiorenne!

Shell

```
>>> anno_di_nascita = 2010  
>>> if anno_di_nascita < 2006:  
    print('Sei maggiorenne!')
```

CONDIZIONI

Shell

```
>>> anno_di_nascita = 2005
>>> if anno_di_nascita < 2006:
    print('Sei maggiorenne!')
```

Sei maggiorenne!

Shell

```
>>> anno_di_nascita = 2010
>>> if anno_di_nascita < 2006:
    print('Sei maggiorenne!')
```

Shell

```
>>> anno_di_nascita = 2010
>>> if anno_di_nascita < 2006:
    print('Sei maggiorenne!')
else:
    print('Sei minorenne!')
```

Sei minorenne!

Altri esempi:

Script

```
anno_di_nascita = 1992

If anno_di_nascita < 2006:
    print('sei Maggioreenne')
elif anno_di_nascita == 2024:
    print('sei appena nato!')
elif anno_di_nascita == 1992:
    print('hai 32 anni!')
else:
    print('sei minorenni ma purtroppo non sei appena nato')
```

Altri esempi:

Script

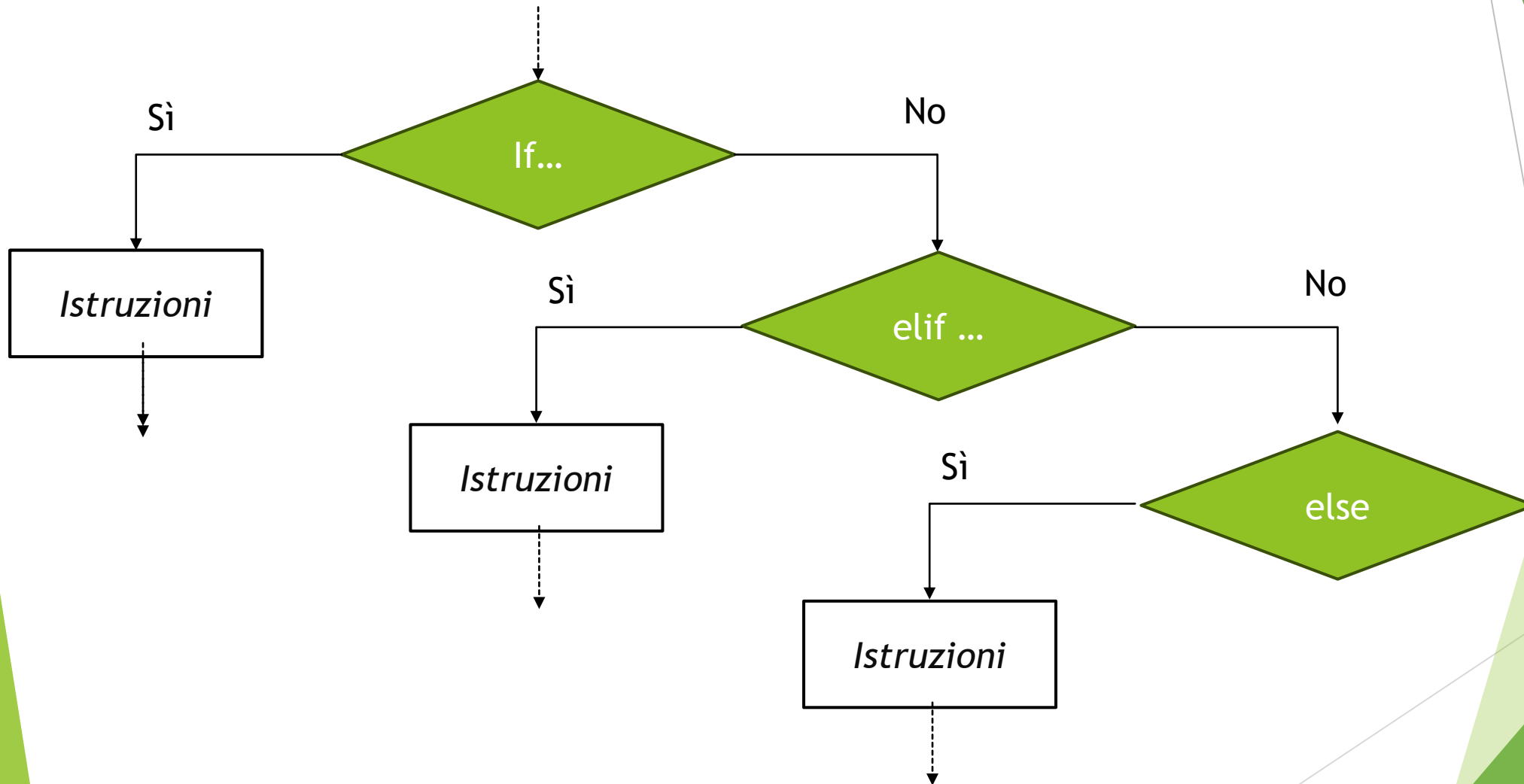
```
anno_di_nascita = 1992

If anno_di_nascita < 2006:
    print('sei Maggioreenne')
elif anno_di_nascita == 2024:
    print('sei appena nato!')
elif anno_di_nascita == 1992:
    print('hai 32 anni!')
else:
    print('sei minorenni ma purtroppo non sei appena nato')
```

Shell

```
Sei Maggioreenne!
```

elif ...



Altri esempi:

Script

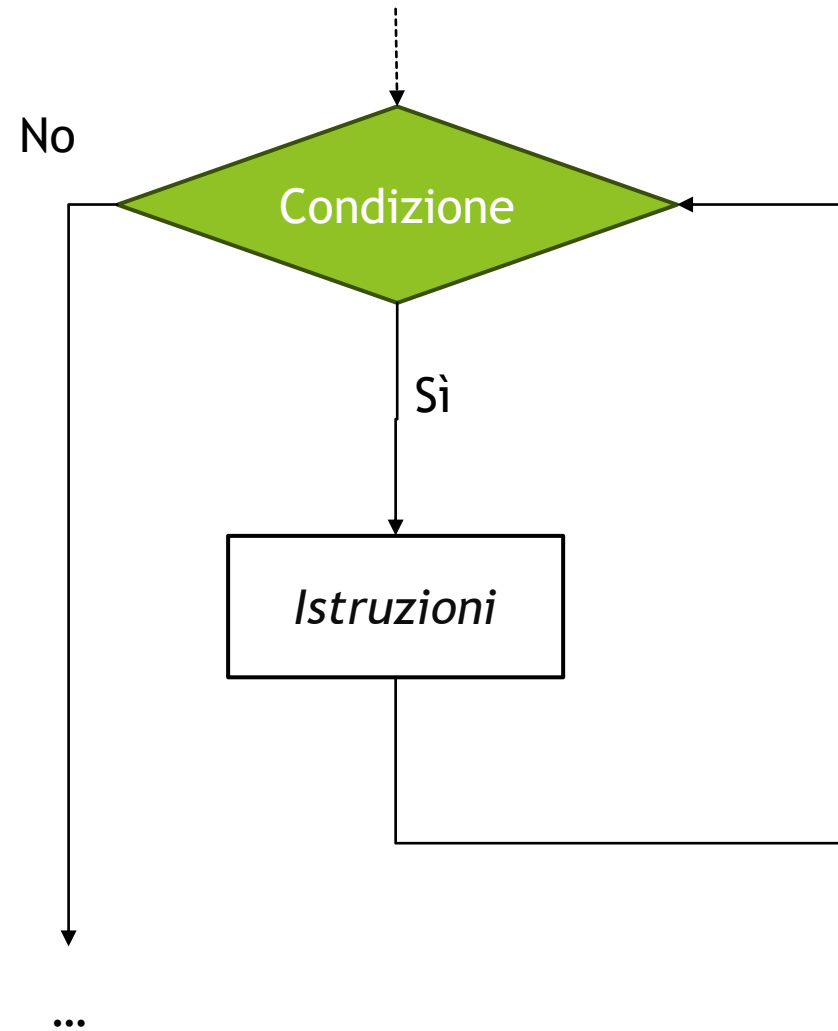
```
anno_di_nascita = 1992
```

```
If anno_di_nascita > 2008 and anno_di_nascita < 2006:  
    print('Hai tra i 16 e i 18 anni!')
```


CICLI

- ▶ Quando ho una parte di codice che si ripete un numero DEFINITO di volte o al persistere di determinate condizioni posso usare i cicli
- ▶ **WHILE...** Questa tipologia di ciclo ripete il codice del blocco sottostante fino al mantenersi di una determinata condizione
 - Le condizioni sono dello stesso tipo del costrutto if
 - Es. While True è un ciclo che non termina mai
- ▶ **For ... in** : questo ciclo ripete il codice nel blocco il numero di volte che definisco io

CICLI



CICLI: while

Script

```
anno_di_nascita = 2050
while anno_di_nascita > 2024:
    anno_di_nascita = input('Non può essere, in che anno sei nato? ')
    anno_di_nascita = int(anno_di_nascita)

eta = 2024 - anno_di_nascita

print('Hai ', eta, 'anni')
```

CICLI: for ... in ...

- La sintassi di for è:

```
for i in range(numero):
```

```
...
```

```
    codice
```

```
...
```

Script

```
n_fiocchi = int(input('Quanti fiocchi di neve vuoi? '))
```

```
for i in range(n_fiocchi):  
    print('*')
```

```
n_scalini = int(input('Quanti fiocchi di neve vuoi? '))
```

```
for i in range(n_scalini):  
    print('|__|')
```

LIBRERIE

Usando import è possibile inserire altri moduli a Python. Ad esempio:

- ▶ RANDOM: tramite questa libreria posso ottenere dei numeri casuali. Es `random.randint(a, b)` mi restituisce un numero random, intero, compreso tra i numeri *a* e *b*
- ▶ MATH: con la libreria *math*, posso usare funzioni `math.sin()`, `math.cos()`, avere la variabile *pi*...
- ▶ TIME: con la libreria *time* posso avere le informazioni sul tempo, per esempio calcolare quanto ci mette un programma ad essere eseguito.
- ▶ TURTLE: libreria grafica che useremo per il nostro progettino di pong

INTRODUZIONE A turtle

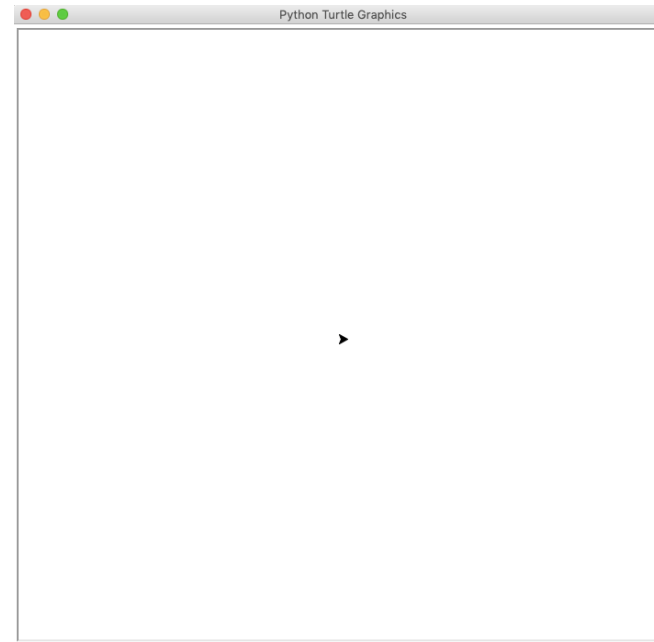


INTRODUZIONE A turtle

- ▶ Libreria per disegnare
- ▶ Molto versatile (interfacce, giochi, grafici...)

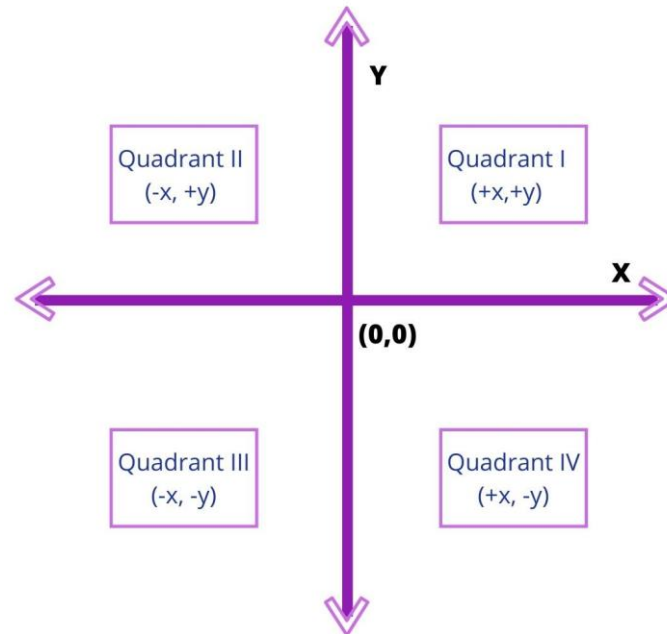
Shell

```
>>> import turtle  
>>> s = turtle.getscreen()  
>>>  
>>>
```



INTRODUZIONE A turtle

- Per disegnare uso la turtle
- Sposto la turtle nel sistema di riferimento



Moving the Turtle

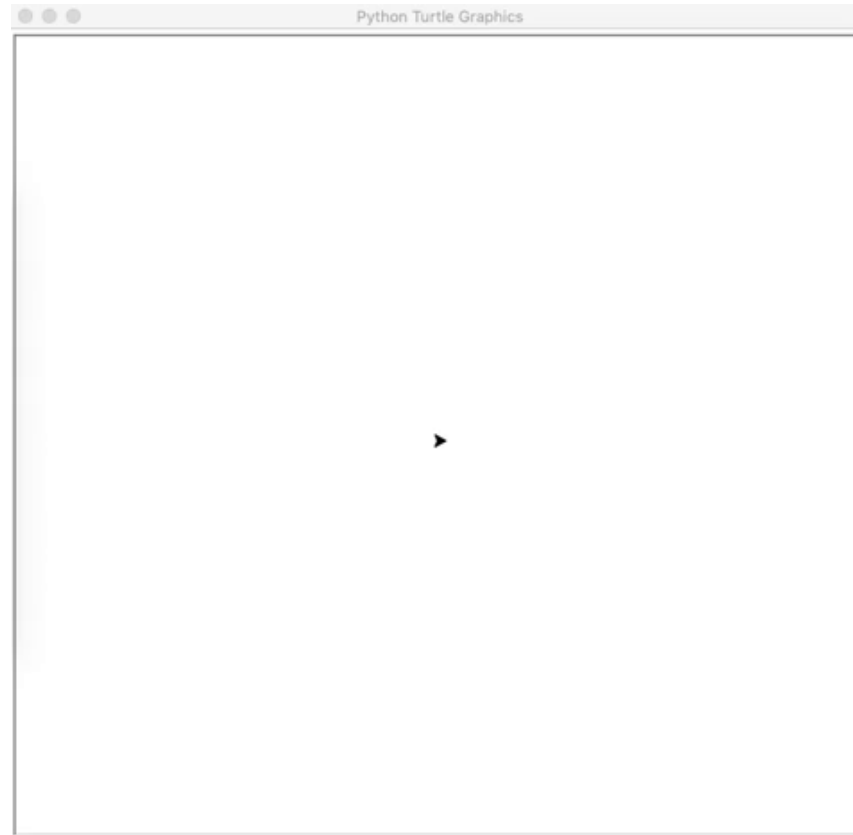
- ▶ Forward (avanti)
- ▶ Backward (indietro)
- ▶ Left (sinistra)
- ▶ Right (destra)

Shell

```
>>> t.right(90)
>>> t.forward(100)
>>> t.right(90)
>>> t.forward(100)
```

Angolo in gradi

Quante unità avanzare

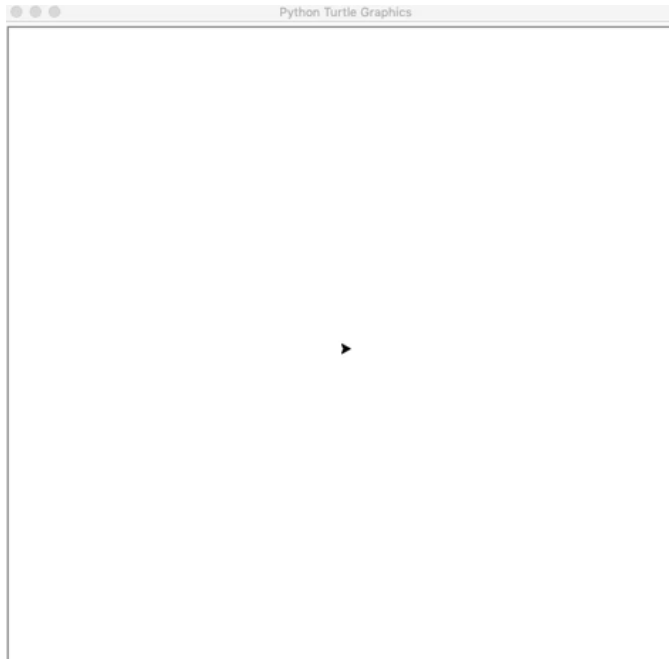


Moving the Turtle

- ▶ Funzione `goto()` sposta in un punto preciso la *turtle*
- ▶ Funzione `home()` riposizione la *turtle* nel punto iniziale
- ▶ La funzione `penup()` ‘alza la penna dal foglio’, `pendown()` ricomincia a disegnare

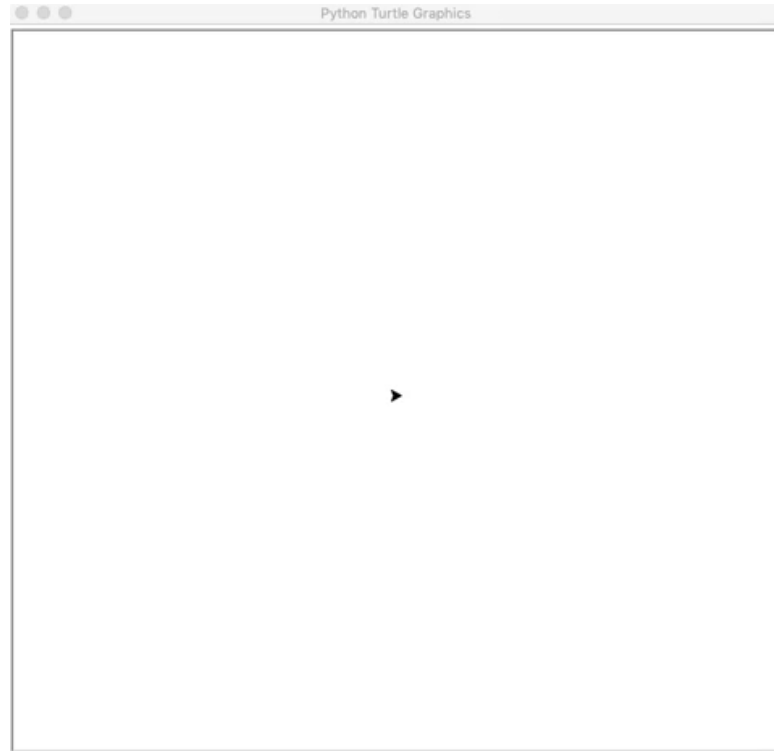
Shell

```
>>> t.goto(100,100)
```



Le coordinate
sono x=100 e
y=100

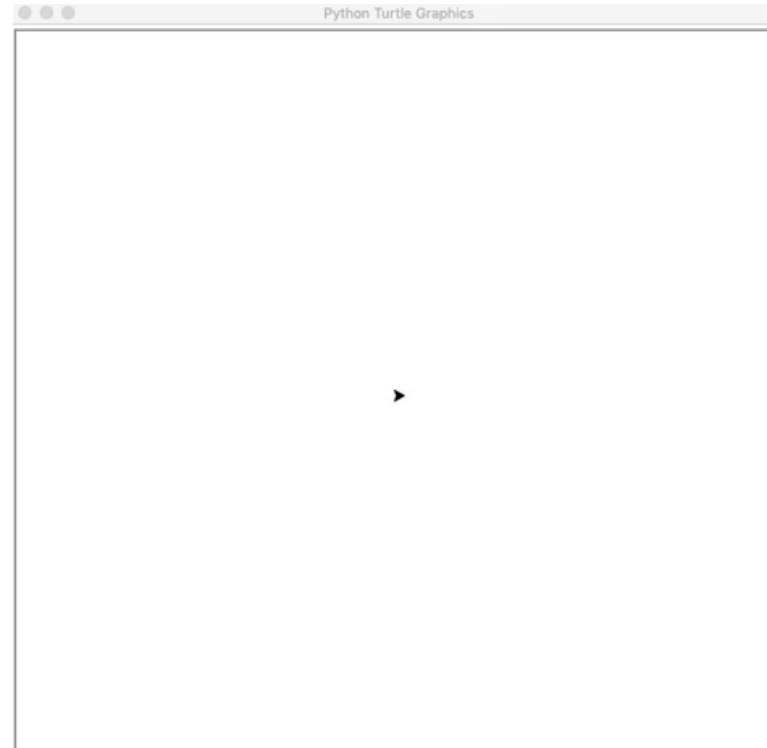
Come disegno un quadrato completo?



Come disegno un quadrato completo?

Shell

```
>>> t.right(90)
>>> t.forward(100)
>>> t.right(90)
>>> t.forward(100)
>>> t.right(90)
>>> t.forward(100)
>>> t.right(90)
>>> t.forward(100)
```

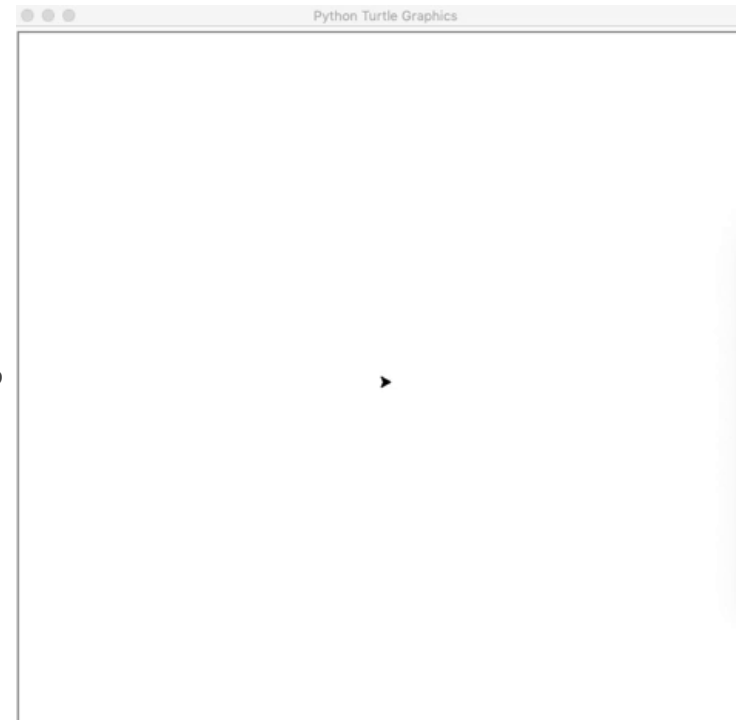


Come disegno un cerchio?

- ▶ Ogni funzione richiede un input diverso
- ▶ `circle()` richiede di inserire il raggio
- ▶ Come posso disegnare molti cerchi concentrici?

Shell

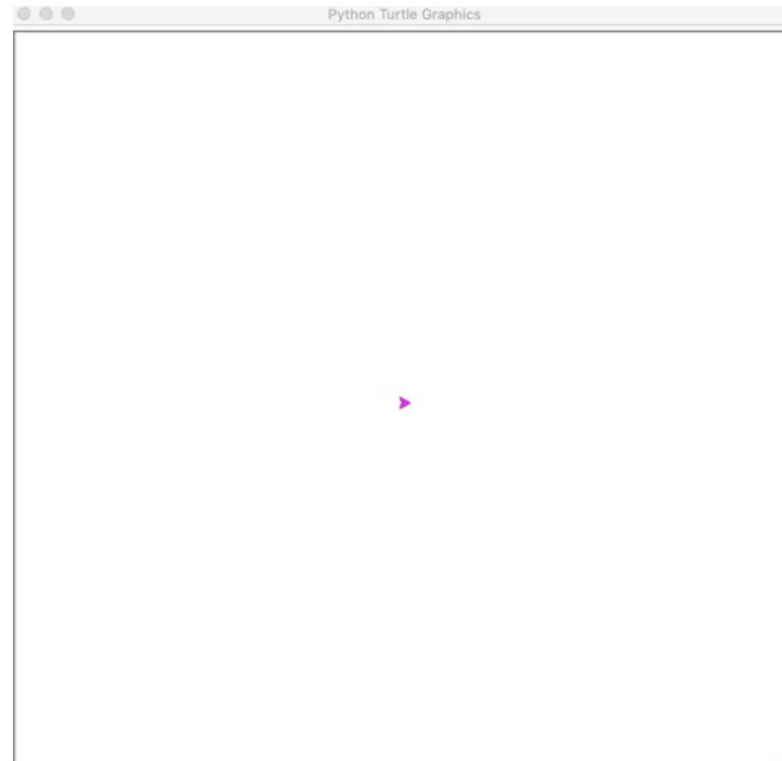
```
>>> turtle.circle(60)
```



Come disegno un cerchio?

Shell

```
>>> turtle.circle(60)  
>>> turtle.circle(40)
```



Funzioni utili in turtle

- ▶ `clearscreen()` 'pulisce la finestra create' e cancella ciò che è stato disegnato
- ▶ `color()` definisco il colore da usare per disegnare (posso usare 'red', 'green', 'yellow', 'blue'...).
- ▶ Alla fine di ogni script mi conviene scrivere `turtle.mainloop()` altrimenti, una volta eseguito, python chiude la finestra in automatico
- ▶ `turtle.screen.title()` mi permette di dare un nome alla finestra di turtle
- ▶ `turtle.screen.bgcolor()` mi permette di cambiare il colore di background della finestra
- ▶ `turtle.teleport(x, y)` trasporta la 'turtle' nella posizione desiderata, senza disegnare

Funzioni utili in turtle

<https://docs.python.org/3/library/turtle.html> Qui trovate
tante altri metodi di questa libreria!

ESERCIZI

- ▶ Trovare tutti i divisori di un numero inserito dall'utente e stamparli a video
- ▶ Fare un check se un numero inserito dall'utente è primo o no
- ▶ Scrivere n numeri della sequenza di Fibonacci, dove n è definito dall'utente. (I primi due numeri sono 1 e 1, i successivi sono la somma dei due numeri precedenti)
- ▶ Chiedere all'utente la data di nascita (anno, mese, giorno). Usare la libreria **datetime** per capire quanti giorni sono passati dal giorno di nascita.

Extra: Se tenessi conto degli anni bisestili?

```
import datetime

oggi = datetime.date.today()

print(oggi.year, oggi.month, oggi.day)

>>> 2024 6 25
```

Esercizi

- ▶ Partendo dal centro di un quadrato e la lunghezza del lato (inseriti in input), fare una funzione che disegni il quadrato.
- ▶ Partendo dai vertici di un triangolo, fare una funzione che disegni il triangolo con *turtle*
- ▶ Costruire quadrati e triangoli in posizioni randomiche usando le funzioni già scritte. Il numero di figure da disegnare dev'essere definito dall'utente
- ▶ Disegnare n cerchi concentrici, dove n è un parametro deciso dall'utente
- ▶ Disegnare una chiocciola secondo la sequenza di fibonacci. `circle()` può avere un argomento aggiuntivo, ovvero quanto voglio disegnare del triangolo, in gradi. Se metto a 360 disegna tutto il cerchio, se metto a 180 ne disegna metà...

ESERCIZI

- ▶ Riprendere l'esercizio per verificare se un punto è dentro a un quadrato e portarlo in python (tramite if annidati)
esercizio aggiuntivo: provare a usare un solo if con le condizioni booleane
- ▶ Riprendere l'esercizio per verificare se un punto è dentro a un cerchio e portarlo in python
- ▶ Se le coordinate non fossero sempre positive?
- ▶ Disegnare con turle le configurazioni dei precedenti esercizi e verificare se i nostri programmi funzionano!