**LA RIPETIZIONE DEFINITA (CICLO FOR)  
SENZA TURTLE**

# 

[INTRODUZIONE](#_4r3c29uxi1t3)

[MODIFICA IL CODICE [10 min]](#_25gotymrs7xg)

[COSA FA IL CODICE? [5 minuti]](#_ga0msgli10m7)

[COSA FA IL CODICE? [5 minuti]](#_qarslsy342q)

[COSA FA IL CODICE? [5 minuti]](#_kgm8l68eoex0)

[DOMANDE e QUIZ [10 min]](#_1vi2gi14r53y)

[ESERCIZI](#_9l3oq24a5etb)

[ESERCIZI AGGIUNTIVI](#_c2k733dsptxh)

# 

# INTRODUZIONE

In questa lezione impareremo ad utilizzare la ripetizione definita **ciclo For**, già introdotta precedentemente, senza la libreria Turtle.  
In particolare, capiremo come passare in rassegna gli elementi di un elenco, che siano numeri o stringhe, assegnandoli uno alla volta ad una variabile, detta **variabile di ciclo**. Per fare ciò approfondiremo ulteriormente la funzione **range()** e utilizzeremo il ciclo for con le **stringhe**.

# 

# 

# MODIFICA IL CODICE [10 min]

**Lavoro in autonomia**

1. [*105-SequenzaNumerica1.py*] Modifica il codice qui sotto in modo tale che venga stampato a video il valore della variabile contatore ad ogni iterazione e non venga disegnato nulla con la libreria Turtle.

import **turtle**

for contatore in **range**(4):

**turtle**.**forward**(100)

**turtle**.**left**(90)

1. [*106-SequenzaNumerica2.py*]. La funzione range() può essere utilizzata anche indicando altri 1 o 2 parametri, modifica il codice precedente in modo tale che ora la funzione range sia:

**range**(1,7)

poi:

**range**(1,10,2)

1. [*107-SequenzaNumerica3.py*] Modifica il codice dell’esercizio 105-SequenzaNumerica1.py (punto 1 di questa pagina) in modo tale che venga richiesto un numero all’utente e vengano stampati a video tutti i numeri da 0 fino al numero inserito.

# 

# COSA FA IL CODICE? [5 minuti]

**Lavoro in coppia**

Di seguito c’è del codice scritto in Python:

numero = int(input("Inserisci un numero: "))

for i in range(1,numero+1):

if (numero % i == 0):  
 print(i, end=" ")

**PS:** In questo esempio la funzione print()è stata utilizzata con un parametro particolare end=" ", il quale permette di stampare senza andare a capo. Infatti, esso sostituisce il carattere di terminazione della riga (di default il ritorno a capo) con il carattere spazio. In questo modo, si aggiunge uno spazio dopo ogni valore stampato e l’output successivo sarà sulla stessa riga.

|  |
| --- |
| Secondo voi, che cosa appare a video eseguendo il codice qui sopra? Descrivete a parole **SENZA eseguire il codice.** |
| L’utente inserisce un numero, |
| Ora copiate e incollate il codice in VSCode (salvate con nome *108\_DivisoriInput.py*) e controllate cosa fa. Fa quello che vi aspettavate? Provate a dare una spiegazione del perché il programma si comporta in questo modo. |
|  |

# 

# COSA FA IL CODICE? [5 minuti]

**Lavoro in coppia**

Di seguito c’è del codice scritto in Python:

for i in range(4):

for j in range(3):  
 print("i:", i, " j:", j)  
 print("- - -")

|  |
| --- |
| Secondo voi, che cosa appare a video eseguendo il codice qui sopra? Descrivete a parole **SENZA eseguire il codice.** |
|  |
| Ora copiate e incollate il codice in VSCode (salvate con nome *109\_CicliAnnidati.py*) e controllate cosa fa. Fa quello che vi aspettavate? Provate a dare una spiegazione del perché il programma si comporta in questo modo. |
|  |

# 

# 

# COSA FA IL CODICE? [5 minuti]

**Lavoro in coppia**

Di seguito c’è del codice scritto in Python:

for c in "ciao":

print(c)

|  |
| --- |
| Secondo voi, che cosa appare a video eseguendo il codice qui sopra? Descrivete a parole **SENZA eseguire il codice.** |
|  |
| Ora copiate e incollate il codice in VSCode (salvate con nome *110\_CaratteriStringa.py*) e controllate cosa fa. Fa quello che vi aspettavate? Provate a dare una spiegazione del perché il programma si comporta in questo modo. |
|  |

# DOMANDE e QUIZ [10 min]

**Lavoro in coppia**

Evidenziate la risposta che ritenete più opportuna per ogni domanda e successivamente controllate il risultato su VSCode.

1. Quanti asterischi vengono stampati?  
   for i in range(5):

for j in range(6):

print("\*")

* 1. 11
  2. 42
  3. 30
  4. 9

1. Quanti asterischi vengono stampati?

for i in range(5):

for j in range(i):

print("\*")

* 1. 10
  2. 25
  3. 16
  4. 15

1. Quale sarà il contenuto della variabile msg dopo l’esecuzione del seguente programma?  
   msg = ""

for i in range(3):

for j in range(2):

msg = msg + "Hip! "

msg = msg + "Urrà! "

* 1. Hip! Hip! Hip! Hip! Hip! Hip! Urrà!
  2. Hip! Hip! Urrà! Hip! Hip! Urrà!
  3. Hip! Hip! Urrà! Hip! Hip! Urrà! Hip! Hip! Urrà!
  4. Urrà!

1. Il seguente codice produrrà un errore?  
   for i in 1,2,3:  
    print(i)

TERMINI CHIAVE DELLA LEZIONE

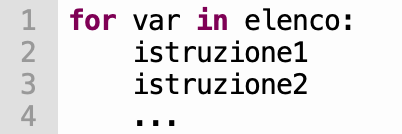
Come abbiamo avuto modo di vedere sia con la libreria Turtle che con Scratch, per ripetere un un blocco di istruzioni abbiamo bisogno della struttura di controllo **iterazione**, detta anche ciclo.

Python mette a disposizione due **costrutti di iterazione**: uno è il costrutto while, che vedremo successivamente, l’altro è il for.

**Spiegazione ciclo for**

L’istruzione for serve per ripetere un blocco di istruzioni per ognuno degli elementi di un elenco. Esso, infatti, permette di passare in rassegna gli elementi di un elenco, assegnandoli uno alla volta a una variabile di ciclo.

La sintassi del ciclo for prevede una variabile di ciclo (var) e un elenco. Le istruzioni da ripetere vanno indentate.



L’elenco può essere formato da:

* numeri  
  for i in 1,2,3:  
   print(i)
* stringhe  
  for i in "Qui","Quo","Qua":  
   print(i)
* altro

ed esso può essere creato per mezzo di apposite funzioni. In particolare, abbiamo utilizzato la funzione range() per generare elenchi di numeri ordinati in modo crescente/decrescente.

**Ciclo for e le stringhe**

In ultimo, abbiamo visto l’utilizzo del ciclo for con le stringhe, utile per scorrere su tutti i caratteri che la compongono.

|  |  |
| --- | --- |
| **print con parametro end=” ”** | |
| **SIGNIFICATO** | **PYTHON** |
| La funzione print()è stata utilizzata con un parametro particolare end=" ", il quale permette di stampare senza andare a capo. Infatti, esso sostituisce il carattere di terminazione della riga (di default il ritorno a capo) con il carattere spazio. In questo modo, si aggiunge uno spazio dopo ogni valore stampato e l’output successivo sarà sulla stessa riga. | for i in range(3):  print(i, end="")  output: 0 1 2  invece di del normale print(i)  output: 0  1  2 |
| **Ciclo for con elenchi di numeri** | |
| **SIGNIFICATO** | **PYTHON** |
| Iterazione su un elenco di numeri generato dalla funzione range().  La funzione range() può essere usate in più modi:   * **range(stop) :** il numero di iterazioni va da 0 a stop non compreso * **range(start, stop)** : il numero di iterazioni va da start a stop non compreso * **range(start, stop, step)** : il numero di iterazioni va da start a stop non compreso con avanzamento uguale a step | for i in range(10):  print(i, end="")  output: 0 1 2 3 4 5 6 7 8 9  for i in range(1,10):  print(i, end="")  output: 1 2 3 4 5 6 7 8 9  for i in range(1,10,2):  print(i, end="")  output: 1 3 5 7 9 |
| **Ciclo for annidato** | |
| **SIGNIFICATO** | **PYTHON** |
| È possibile annidare un ciclo for in un altro ciclo for.  In questo caso, la variabile cambia il valore solo ogni volta che è stato completato il ciclo for interno.  Abbiamo già visto questo meccanismo con turtle, ad esempio nell’esercizio *79\_EsagonoDiTriangoliCiclo.py* | for i in range(4):  for j in range(3):  print("i:", i, " j:", j)  print("---")  i: 0 j: 0  i: 0 j: 1  i: 0 j: 2  - - -  i: 1 j: 0  i: 1 j: 1  i: 1 j: 2  - - -  i: 2 j: 0  i: 2 j: 1  i: 2 j: 2  - - -  i: 3 j: 0  i: 3 j: 1  i: 3 j: 2 |
| **Ciclo for con stringhe** | |
| **SIGNIFICATO** | **PYTHON** |
| È possibile scorrere i caratteri di una stringa con il ciclo for.  La variabile contatore conterrà ogni singolo carattere. | for c in "ciao":  print(c)  output: c  i  a  o |

# 

# 

# ESERCIZI

**Lavoro in autonomia**

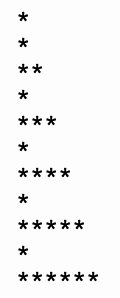
1. [*111\_Tabellina5.py*] Visualizzare la tabellina del 5.
2. [*112\_AllaSeconda.py*] L'utente inserisce un numero N, visualizzare tutti i numeri fino a N elevati alla seconda.
3. [*113\_QuadratiTraAeB.py*] Scrivere tutti i quadrati dei numeri compresi tra due numeri dati dall'utente.
4. [*114\_SommaN.py*] Calcolare la somma dei primi N numeri naturali dove N è un numero inserito dall'utente).
5. [*115\_SommaPositiviNegativi.py*] Calcolare separatamente la somma dei numeri positivi e quelli negativi in una lista di numeri di lunghezza N (inserito anch'esso dall'utente).
6. [*116\_SommaNumeriCasuali.py*] Chiedere all’utente un numero N, generare casualmente N numeri compresi tra 0 e 100 e calcolarne la somma.
7. [*117\_Potenze.py*] L'utente inserisce due numeri: *base*, *esponente*. Visualizzare tutti i numeri la cui base è inserita dall’utente e che hanno come potenza tutti i numeri da 0 fino all'esponente scelto dall'utente.

(esempio: inserendo 3 come base e 4 come esponente visualizza 1,3,9,27,81).

1. [*118\_Primo.py*] Stabilire se il numero inserito dall'utente è un numero primo(ricorda che un numero primo è divisibile solo per 1 e per sé stesso!)
2. [*119\_Multipli.py*] Visualizzare in ordine decrescente i multipli di un numero inserito dall'utente, partendo da un valore massimo anch'esso impostato dall'utente. Ad esempio inserendo 3 e 12 visualizza 12 9 6 3.
3. [120*\_FrequenzaA.py*] Chiedi in input all’utente una stringa e conta quante volte compare la lettera a (sia maiuscolo che minuscola). Mostra a video il risultato.

# ESERCIZI AGGIUNTIVI

**Lavoro in autonomia**

1. [*121-Coppie20.py*] Scrivi un programma che stampa tutte le coppie di numeri naturali (interi positivi) la cui somma è 20.
2. [*122-Intervallo.py*] Scrivi un programma che riceva in input due interi (a e b) e calcoli la somma di tutti i numeri compresi nell’intervallo [a,b] (estremi inclusi). Il programma deve stampare un messaggio di errore se b < a.  
   Per esempio, se l’utente inserisce 3 e 6, il programma deve stampare 18.
3. [*123-Cubi.py*] Scrivi un programma che riceva in input un intero n e stampi tutti gli interi che sono cubi di un intero e sono minori o uguali a n.  
   **Per esempio**, se n= 100, il programma deve stampare 1 8 27 64.
4. [*124-MassimoNumero.py*] Data in input una sequenza di numeri, determinare il numero massimo della sequenza di numeri inserita. Le informazioni che non hai chiedile all’utente, ad esempio quanti e quali sono i numeri che compongono la sequenza. Suggerimento: acquisisci la sequenza di numeri con un ciclo for.
5. [*125-TavolaPitagorica.py*] Mostrare a video la tavola pitagorica dei numeri fino al 10 compreso. Nella figura qui sotto trovi un esempio di tavola pitagorica.   
   
6. [*126-PariDispari.py*] Scrivi un programma che, scorrendo i numeri da 1 a 11 compreso, stampa un solo asterisco se il numero è pari. Mentre, se il numero è dispari, stampa un numero di asterischi che aumenta di 1 ad ogni numero dispari.  
   Ad esempio, quando il numero è 1 l’output sarà “ \* ”, quanto è 2 l’output sarà “\*”, quando è 3 l’output sarà “ \*\* ”, quando è 4 sarà “ \* ”, quando è 5 sarà “ \*\*\* ”, ecc..  
   L’output lo trovi nella figura qui sotto:
7. [*127-RandomBarcode.py*] Scrivi un programma che genereri una sequenza di | seguendo il risultato del lancio della moneta: se esce testa mostro una “|”, se esce croce mostro uno spazio “ ”.  
   **Suggerimento:** puoi simulare il lancio della moneta con il random e considerare 0=testa e 1=croce.Esempio:  
   
8. [*128-Piramide.py*] Prova a replicare la piramide qui sotto.  
   