LABORATORIO DI PYTHON

ITERAZIONE IN PYTHON

22 Marzo 2019

Correzione esercizi per oggi

ESERCIZIO 1 PER CASA

Scrivere una funzione che prende come parametri due tuple `t1` e `t2` e un numero intero `n`. La prima tupla è da modificare aggiungendo tutti i valori della seconda tupla alla posizione `n`. La funzione restituisce la tupla aggiornata. Es. se `t1` $= (1, 3, 2)$, `t2` $= (9, 7)$, `n` $= 1$ restituirà $(1, 9, 7, 3, 2)$.

ESERCIZIO 1 PER CASA

Scrivere una funzione che prende come parametri due tuple `t1` e `t2` e un numero intero `n`. La prima tupla è da modificare aggiungendo tutti i valori della seconda tupla alla posizione `n`. La funzione restituisce la tupla aggiornata. Es. se `t1` $= (1, 3, 2)$, `t2` $= (9, 7)$, `n` $= 1$ restituirà $(1, 9, 7, 3, 2)$.

```
1 def inserisci(t1, t2, n):
2     return t1[:n]+t2+t1[n:]
```

ESERCIZIO 2 PER CASA

Scrivere un programma che prese due stringhe in input le stampi in ordine alfabetico, oppure ne stampi solo una delle due se sono identiche.

ESERCIZIO 2 PER CASA

Scrivere un programma che prese due stringhe in input le stampi in ordine alfabetico, oppure ne stampi solo una delle due se sono identiche.

```
1  s1 = input("Prima stringa: ")
2  s2 = input("Seconda stringa: ")
3
4  if s1 < s2:
5      print(s1, s2)
6  elif s1 > s2:
7      print(s2, s1)
8  else:
9      print(s1)
```

ESERCIZIO 3 PER CASA

Scrivere una funzione che stampi a video

- *le consonanti* della stringa passata come parametro
- il *numero di vocali* della stessa stringa

La funzione **non stampa e non conteggia** spazi, punteggiatura, simboli vari...

ESERCIZIO 3 PER CASA

Scrivere una funzione che stampi a video

- *le consonanti* della stringa passata come parametro
- il *numero di vocali* della stessa stringa

La funzione **non stampa e non conteggia** spazi, punteggiatura, simboli vari...

```
1  import string
2  def cons_n_voc(s):
3      vocali = "aeiouAEIOU"
4      n_vocali = 0
5      print("Consonanti: ")
6      for l in s:
7          if l in vocali:
8              n_vocali = n_vocali + 1
9          elif l in string.ascii_letters: #so gia' che non e' una
       vocale
10             print(l)
11     print("Numero di vocali:", n_vocali)
```

Turtle

MODULO `turtle`

- Immaginiamo di essere una tartaruga che cammina su un piano cartesiano.
- Sotto la pancia abbiamo una penna, che lascia dunque una traccia mentre camminiamo.
- Possiamo muoverci e ruotare (o anche teletrasportarci su un punto specifico del piano).
- Possiamo anche sollevare la penna per non lasciare il segno.

NB: **non** salvare il file come `turtle.py`

MODULO `turtle`

- Immaginiamo di essere una tartaruga che cammina su un piano cartesiano.
- Sotto la pancia abbiamo una penna, che lascia dunque una traccia mentre camminiamo.
- Possiamo muoverci e ruotare (o anche teletrasportarci su un punto specifico del piano).
- Possiamo anche sollevare la penna per non lasciare il segno.

NB: **non** salvare il file come `turtle.py`

Cosa accade?

```
1 import turtle
2
3 turtle.forward(100)
```

MODULO `turtle`: FUNZIONI

- `turtle.forward(distance)`
- `turtle.backward(distance)`
- `turtle.right(angle)`
- `turtle.left(angle)`
- `turtle.pendown()`
- `turtle.penup()`
- `turtle.goto(x, y)`
- `turtle.setheading(to_angle)`

- e molte altre:
  `https://docs.python.org/3/library/turtle.html`

ESERCIZIO

Disegnare, procedendo in senso antiorario, un quadrato di lato 100 con la tartaruga, *senza usare il* `for`.

ESERCIZIO

```
1 import turtle
2 turtle.forward(100)
3 turtle.left(90)
4 turtle.forward(100)
5 turtle.left(90)
6 turtle.forward(100)
7 turtle.left(90)
8 turtle.forward(100)
```

Potrebbe aver senso usare un for

INTERMEZZO: RIPETI n VOLTE

Posso usare i range (che vedremo meglio in seguito) per far sì che un'istruzione venga eseguita un numero determinato (finito) di volte.

Cosa stampa questo codice?

```
1 for i in range(4):
2     print("Ciao")
```

ESERCIZIO

Disegnare, procedendo in senso antiorario, un quadrato di lato 100 con la tartaruga. Usare il for

ESERCIZIO

```
1 import turtle
2 for i in range(4):
3     turtle.forward(100)
4     turtle.left(90)
```

ESERCIZIO

Usando il ciclo for:

- Scrivere un programma per disegnare un triangolo regolare.
- Poi per disegnare un pentagono regolare.
- Poi per disegnare una stellina a 5 punte: 

ESERCIZIO

Triangolo:

```
1 for i in range(3):
2     turtle.forward(100)
3     turtle.left(120) #angolo esterno!
```

Pentagono:

```
1 for i in range(5):
2     turtle.forward(100)
3     turtle.left(72)
```

Stellina:

```
1 for i in range(5): #ho 5 lati
2     turtle.forward(100)
3     turtle.right(144) #doppio del pentagno: salto un vertice
```

ESERCIZIO

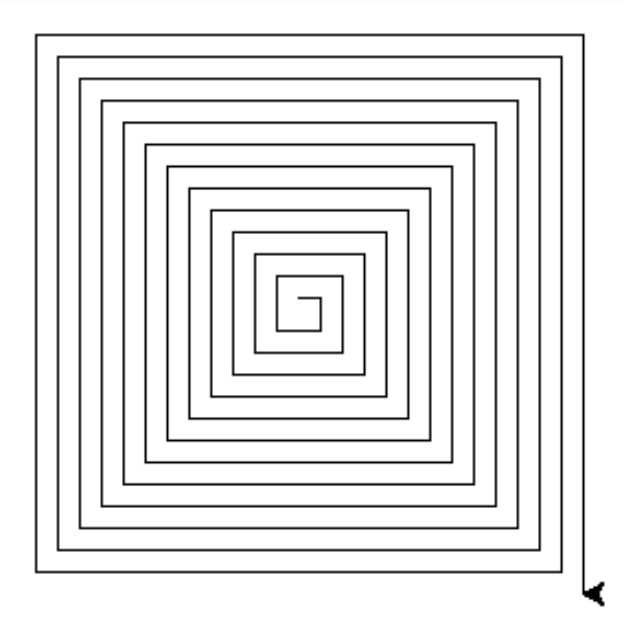
Scrivere una funzione che prende come parametri: il numero di lati e la lunghezza del lato e disegna il poligono regolare corrispondente. Controllare che il numero di lati sia maggiore di 2.

ESERCIZIO

```
1 import turtle
2 def poligono_regolare(n, l):
3     if n>2:
4         for i in range(n):
5             turtle.forward(l)
6             turtle.left(360/n)
```

ESERCIZIO

Scrivere un programma che disegna una spirale quadrata.



ESERCIZIO

```
1 import turtle
2 dist = 10
3 for i in range(50):
4     turtle.forward(dist)
5     turtle.right(90)
6     dist = dist+5
```

INTERMEZZO: `range`

Tipo di sequenza immutabile.

Rappresentano intervalli sui numeri interi

- `range(n)` rappresenta l'intervallo su interi $[0, n[$
- `range(a,b)` rappresenta l'intervallo su interi $[a, b[$
- `range(a, b, s)`, se $s > 0$, rappresenta l'intervallo su interi $[a, a+s, a+2s, ..., a+is]$ in cui $a + is < b$
- `range(a, b, s)`, se $s < 0$, rappresenta l'intervallo su interi $[a, a+s, a+2s, ..., a+is]$ (n.b. s negativo!) in cui $a + is > b$

INTERMEZZO: `range`

Tipo di sequenza immutabile.

Rappresentano intervalli sui numeri interi

- `range(n)` rappresenta l'intervallo su interi $[0, n[$
- `range(a,b)` rappresenta l'intervallo su interi $[a, b[$
- `range(a, b, s)`, se $s > 0$, rappresenta l'intervallo su interi $[a, a+s, a+2s, ..., a+is]$ in cui $a+is < b$
- `range(a, b, s)`, se $s < 0$, rappresenta l'intervallo su interi $[a, a+s, a+2s, ..., a+is]$ (n.b. s negativo!) in cui $a+is > b$

Per vedere i range in forma estesa, possiamo convertirli in tuple

```
>>> tuple(range(10))
(0, 1, 2, 3, 4, 5, 6, 7, 8, 9)
>>> tuple(range(1,6))
(1, 2, 3, 4, 5)
>>> tuple(range(0,30,5))
(0, 5, 10, 15, 20, 25)
>>> tuple(range(1,10,2))
(1, 3, 5, 7, 9)
```

```
>>> tuple(range(1,10,-2))
()
>>> tuple(range(10,1,-2))
(10, 8, 6, 4, 2)
>>> tuple(range(0))
()
>>> tuple(range(1,0))
()
```

Uso dell'indice dell'iterazione

INDICE NEL FOR

Cosa stampa questo codice?

```
1 for i in range(100):
2     print(i)
```

INDICE NEL FOR

Cosa stampa questo codice?

```
1 for i in range(100):
2     print(i)
```

E questo?

```
1 for i in range(1,101):
2     print(i)
```

INDICE NEL FOR

Cosa stampa questo codice?

```
1 for i in range(100):
2     print(i)
```

E questo?

```
1 for i in range(1,101):
2     print(i)
```

E questo?

```
1 for i in range(0,100,2):
2     print(i)
```

ESERCIZIO

Scrivere una funzione che prende come parametro un numero `n` e ne stampa la tabellina (da 0 a 10). Esempio `tabellina(2)` stampa:

```
2x0=0
2x1=2
2x2=4
2x3=6
2x4=8
2x5=10
2x6=12
2x7=14
2x8=16
2x9=18
2x10=20
```

ESERCIZIO

```
1 def tabellina(n):
2     for i in range(0,11):
3         print(n,"x",i,"=",n*i,sep='')
```

ESERCIZIO

Scrivere una funzione che stampa il Triangolo di Floyd di dimensione n, passato come parametro. L'i-esima riga del triangolo è una tupla di i numeri naturali. Es. con $n = 5$ il triangolo sarà

```
(1,)
(2, 3)
(4, 5, 6)
(7, 8, 9, 10)
(11, 12, 13, 14, 15)
```

Suggerimento: notare che il primo elemento di ogni riga corrisponde al numero della riga precedente (partendo a contare le righe da 1) sommato al primo elemento della riga precedente.

ESERCIZIO

```
1  def Floyd(n):
2      pe = 1 #primo elemento
3      #for per ogni "riga" del triangolo (da 1 a n)
4      for riga in range(1,n+1):
5          #primo elemento riga successiva
6          pes = pe + riga
7          #ricorda: ultimo elemento escluso
8          print(tuple(range(pe, pes)))
9          #aggiorno prima di passare alla riga succ.
10         pe = pes
```

Esercizi per casa

ESERCIZI PER CASA

1. Scrivere una funzione che restituisce `True` se una stringa passata come parametro è palindroma (senza considerare gli spazi, la punteggiatura ma facendo distinzione tra maiuscole e miniscole: es "`Anna`" **non** è palindroma, "`ANGOLO BAR, A BOLOGNA!`" sì). Usare il ciclo `for`.
2. Scrivere una funzione che presi due numeri come parametri della funzione, restituisca il Massimo Comun Divisore (MCD) tra i due numeri. Usare il ciclo `for`.
3. Scrivere una funzione che preso come parametro un numero `n` (controllare che `n>2`) restituisce il più piccolo `c` (`c>=2`) tale che `MCD(n,c) == 1`. Usare il ciclo `for` e la funzione dell'esercizio 2.
4. Scrivere una funzione con un parametro `n`. Se `n>=7`, disegna una "stellina" a `n` punte. Si tratta di una generalizzazione della versione a 5 punte. NB! Il numero dei lati da "saltare" può essere scelto come un numero **coprimo con `n`** (e dunque si può usare la funzione dell'esercizio 3).