LABORATORIO DI PYTHON

ESERCIZI VARI, PRATICHE DI DEBUG E DI TRACING

29 Marzo 2019

Debug

ESERCIZIO A – COSA NON VA IN QUESTO CODICE?

Scrivere una funzione che prende come parametro una tupla `t` e restituisce `True` se tutti i valori sono in ordine **strettamente crescente**, `False` altrimenti.

```
1  def crescente(t):
2      for i in range(len(t)):
3          if t[i+1]>t[i]:
4              return True
5      return False
6
7  print(crescente((1,2,3,4,5,6))) # Atteso: True
8  print(crescente((1,2,2,4))) # Atteso: False
9  print(crescente((7,1,4,5,3))) # Atteso: False
10 print(crescente((1,))) # Atteso: True
11 print(crescente(())) # Atteso: True
```

ESERCIZIO B

Scrivere una funzione che restituisce una tupla contenente tutti i divisori propri (`n` escluso) di un numero naturale `n` preso come parametro.

N.B. Un numero è perfetto se il numero è uguale alla somma dei divisori propri.

Scrivere una funzione che prende come parametro un numero naturale `n` e restituisce `True` se il numero è un numero perfetto, `False` altrimenti. Usare la funzione creata precedente.

Esempi di numeri perfetti sono il 6, 28, 496, 8128, 33550336.

ESERCIZIO B – COSA NON VA IN QUESTO CODICE?

```
1  def divisori_propri(n):
2      divisori = ()
3      for i in range(n):
4          if n%i = 0:
5              divisori += i
6
7  def numero_perfetto(n):
8      divisori = divisori_propri(n)
9      somma_divisori = 0
10     for divisore in divisori:
11         somma_divisori += divisore
12         if somma_divisori == n:
13             return True
14         else:
15             return False
```

ESERCIZIO C – COSA NON VA IN QUESTO CODICE?

Un plateau è una sottosequenza di almeno due elementi contigui con lo stesso valore. Scrivere una funzione `plateau(t)` che, data una tupla `t`, restituisce la tupla degli elementi distinti di `t` che sono valori di un plateau.

Esempio `plateau((3,3,0,2,2,2,0,3,3,4,4))` restituisce `(3,2,4)`.

```
1  def plateau(t):
2      for i in range(len(t-1)):
3          if t[i] == t[i+1] and risultato[-1] != t[i]:
4              risultato += t[i]
5          return risultato
```

ALTRI ESERCIZI, EVENTUALMENTE DA FINIRE A CASA.

Scrivere una funzione `confronta(T1, T2)` che prese due stringhe `T1` e `T2` restituisce una tupla contenente gli `i` tali che `T1[i]` è uguale ad uno dei caratteri `T2[i-1]`, `T2[i]`, `T2[i+1]`.

Esempio.

```
1 print(confronta('asca', 'lasca')) #(0,1,2,3)
2 print(confronta('la mamma in', 'la nonna ti')) #(0,1,2,7,8,9)
3 print(confronta('acca', 'zonzo')) #()
```

ALTRI ESERCIZI, EVENTUALMENTE DA FINIRE A CASA.

Giulio Cesare era solito criptare i suoi messaggi sostituendo a ogni lettera quella corrispondente dell'alfabeto "spostato in avanti".

Scrivere due funzioni:

- Una funzione che ritorna l'indice del carattere `c` nella stringa `s`, o l'indice della prima volta che compare `c`, se compare più volte, o None se `c` non è presente
- Una funzione che "`modifica`" la stringa `s` presa come parametro sostituendo a ogni lettera la lettera che si trova 13 posizioni piu' avanti nell'alfabeto. Esempio `'b'` diventa `'o'`, `'m'` diventa `'z'`, `'n'` diventa `'a'`. Supponiamo di lavorare solo con alfabeto minuscolo, convertendo eventualmente `s` in tuttominuscolo. Tutti gli altri caratteri (cifre, punteggiatura, spazi) non vengono modificati.

N.B. Esiste la costante `string.ascii_lowercase` nel modulo string che contiene tutti e soli i caratteri dell'alfabeto (senza numeri e punteggiatura).

ALTRI ESERCIZI, EVENTUALMENTE DA FINIRE A CASA.

Date due tuple di naturali tutti distinti A e B, diciamo che costituiscono un involucro se una delle due compare come sottosequenza contigua dentro l'altra, con almeno un elemento a sinistra e almeno un elemento a destra che non le appartengono.

Esempio. (0,1,2,3,11,16) e (1,2,3) costituiscono un involucro; (1,2,3) e (1,2,3,11) non lo sono.

Scrivere una funzione `involucro(A,B)` che, presi come parametri due tuple di naturali restituisce `True` se e solo se esse costituiscono un involucro.

Test. Usare il seguente codice per testare la funzione con un input.

```
1  A = eval(input("Inserisci una tupla: "))
2  B = eval(input("Inserisci una tupla: "))
3  print(A,B) #verifico se l'input e' corretto
4  print(involucro(A,B))
```