Esercitazione 5

06 dicembre 2017

Termine per la consegna dei lavori: **martedì 12 dicembre** ore **23.55**.

Istruzioni

Ogni esercizio dovrà esser salvato in un file con estensione `.py` e consegnato tramite Moodle su sito `https://elearning.unipd.it/math`, sezione "**Matematica: Laurea Triennale**" corso di "**Laboratorio di Programmazione**" cliccando su "**Esecitazione5**".
È obbligatorio che all'interno di ogni file sia riportato il vostro nome, cognome e numero di matricola (potete riportarli all'interno di una riga commento all'inizio del file, es: `#Mario Rossi 1234567`).
Controllate che l'esecuzione del comando:

```
python <nome_file>.py
```

per ognuno degli esercizi produca l'output desiderato.

ATTENZIONE!

Gli unici moduli importabili ammessi in questa esercitazione sono i moduli `math` e `random`. Esercizi risolti utilizzando altri moduli importati riceveranno il punteggio minimo. Python contiene delle **built-in functions**, che potete utilizzare e la cui lista si può trovare a questo indirizzo: `https://docs.python.org/2/library/functions.html`.

Esercizio 1

Scrivere la funzione iterativa `f(n)` dove $n \geq 0$ intero, che ritorna il numero di passi necessari per trasformare il numero n secondo questo semplice processo iterativo: se n è composto da una unica cifra allora termina, altrimenti $n =$ `digit_sum(n)`, dove `digit_sum` è una funzione di supporto che restituisce la somma delle cifre del numero, ad esempio: `digit_sum(734)` = 7+3+4 = 14.

Esempio: partendo dal numero 734 si avrà:
$734 \rightarrow 7 + 3 + 4 = 14 \rightarrow 1 + 4 = 5$, con 2 passi.

Esercizio 2

Scrivere una funzione iterativa `HipHurra` che dato `n` (intero) stampa `n` volte "Hip Hurra" nel modo seguente: fissato ad esempio n=3 la funzione dovrà stampare: "Hip Hip Hip Hurra Hurra Hurra".

Esercizio 3: Il gioco del 15

Link di spiegazione del gioco: `http://it.wikipedia.org/wiki/Gioco_del_quindici`
Tramite l'utilizzo di funzioni il programma deve:

1. Mostrare la tabellina di gioco disordinata;
2. Chiedere all'utente quale mossa vuole effettuare;
3. Verificare se effettivamente la mossa sia corretta e nel caso affermativo modificare la tabella di gioco, altrimenti ripetere la richiesta;
4. Verificare se si è raggiunto lo stato finale (tab. ordinata), altrimenti ripetere tutti i passi dal punto 1.

Consiglio: per "disordinare" in modo corretto (quindi risolvibile) la tabella iniziale del gioco del 15, si possono effettuare delle mosse scelte in modo casuale ed effettuarle solo se sono mosse corrette.

Esercizio 4: distanza minima

Scrivere la funzione `distance(L1,L2)`, la quale prende in input due liste di interi, e restituisce la distanza minima tra i valori di L1 ed i valori di L2, ovvero restituisce la *differenza* minima in valore assoluto tra i valori di tutte le coppie (l_1, l_2) con $l_1 \in L1$ e $l_2 \in L2$.
Esempio, con $L1 = [5, 9, 11]$ ed $L2 = [8, 19]$, la distanza minima è 1, ed è ottenuta come differenza tra i valori della coppia (9,8).

Esercizio 4:

Sia X una lista di coppie $[(y_1, x_1), \ldots, (y_n, x_n)]$ dove $x_i \in \mathbb{R}_+$ e $y_i \in C = \{c_1, \ldots, c_k\}$ è la *classe* di x_i.
Scrivere la funzione `find_classes(X)` che, dato in input la lista X, calcola e ritorna la lista di tutte le possibili classi $C = [c_1, \ldots, c_k]$ in X.
Scrivere inoltre la funzione `find_indices(X,C)` avente le liste X e C in input, che restituisca il dizionario I dove ad ogni classe c_j viene associata la lista degli indici delle coppie (y_i, x_i) in X per cui $c_j = y_i$.
Esempio, con $X = [('a', 6), ('a', 1), ('b', 1), ('b', 50), ('b', 11), ('c', 33)]$, C sarà $['a', 'b', 'c']$, mentre I:

```
{
  'a':[0,1],
  'b':[2,3,4],
  'c':[5]
}
```

(ricorda che le classi in X non sono necessariamente ordinate come nell'esempio).

Una volta ottenuto C ed I, realizzare il dizionario D dove le chiavi sono tutte le coppie ottenute da *combinazioni* tra classi (c_r, c_s) ed i valori sono la distanza minima tra gli elementi della classe c_r e gli elementi della classe c_s. Utilizzare la funzione `distance` dell'esercizio precedente. Continuando l'esempio, D sarà:

```
{
  ('a','b'):0,  # la distanza minima e' data dalle coppie ('a',1) e ('b',1)
      con indici 1 e 2 in X
  ('a','c'):27, #  tra le coppie ('a',6) e ('c',33) con indici 0 e 5 in X
  ('b','c'):17,  #  tra le coppie ('b',50) e ('c',33) con indici 4 e 5 in X
}
```

ATTENZIONE: per realizzare D bisognerà sfruttare I e C, ricordando però che I contiene gli indici degli elementi, mentre la funzione `distance` richiede i valori degli elementi nelle liste in input. A tal proposito si consiglia di utilizzare una funzione di supporto oppure un descrittore di lista.