Sapienza Università di Roma
Corso di Laurea in Ingegneria Informatica e Automatica A.A. 2019-20

Esercitazione Python n. 10 -- 3 dicembre 2019

Obiettivo dell'esercitazione è esercitarsi nella lettura di file e nell'utilizzo di espressioni regolari e dizionari.
All'intero dell'esercitazione vi sono cartelle relative ad ogni singolo esercizio, denominate EsercN, al cui interno sono definiti i file .py e gli eventuali file di input relativi allo specifico esercizio.
I file .py incorporano un codice di test per le vostre funzioni.
Per ogni esercizio, aprite il file .py relativo e modificate SOLO il contenuto della funzione. Eseguendo il file .py si otterrà il responso del test sulla console.

Esercizi

- **Ex1(file)**: scrivere la funzione Python che, preso in ingresso il nome di un **file** di testo calcoli, usando le espressioni regolari, quante sono le sequenze non sovrapposte di 2 parole consecutive aventi la seguente proprietà:

  "La due parole hanno la stessa lettera iniziale, finale ed una terza lettera in comune."

  Ad esempio, prendendo come input il file contenente il seguente testo:

  `tanto va **Al**d**i**n**o** **a**nnaspan**do** in giro che era andato a casa`

  la funzione deve restituire come risultato `2`.

  Notate che le sequenze di parole richieste non devono essere sovrapposte. Ad esempio, se il file contiene il testo:

  `ossa orma ocra ossa`

  la funzione deve restituire come risultato `2` (le due sequenze sono `ossa orma` e `ocra ossa`)

  Usate il flag `re.IGNORECASE` per non fare differenza tra maiuscole e minuscole. Potete usare le funzioni `re.finditer()` o `re.findall()` a vostra scelta. Se usate la funzione `re.findall()`, per contare il numero di soluzioni trovate basta usare la funzione `len()` applicata al risultato della `re.findall()`. Una parola è una sequenza di caratteri alfabetici `([a-z])`. Ignorate le lettere accentate, che non saranno presenti nel file.

- **Ex2(file):** scrivere la funzione Python che, preso in ingresso il nome di un **file** di testo calcoli, usando le espressioni regolari, quante volte compaiono in una stessa riga tre parole consecutive (senza sovrapposizioni) tutte con la stessa doppia.

  Ad esempio, prendendo come input il file contenente il seguente testo:

  `va Aldo oziando da**ll**a sta**ll**a a**ll**a casa che chiedeva da solo e va via da casa`

  la funzione deve restituire come risultato `1`.

  Adottate le stesse assunzioni descritte per l'esercizio Ex1 e seguite gli stessi suggerimenti.

- **Ex3(file)**: scrivere una funzione Python che, preso in ingresso il nome di un **file** di testo calcoli, usando le espressioni regolari, quante volte compare una parola con la seguente proprietà:

  > "la lettera iniziale e finale della parola sono uguali ed all'interno della parola compare almeno una doppia."

  Si noti che la doppia deve comparire all'interno della parola, senza quindi considerare il primo e l'ultimo carattere della parola stessa. Ad esempio, prendendo come input il file contenente il seguente testo:

  `tanto` **`attacca`** `contro` **`elettore`** `in giro` **`abbastanza`** `era andato a casa`

  la funzione deve restituire come risultato `3`.

  Adottate le stesse assunzioni descritte per l'esercizio Ex1 e seguite gli stessi suggerimenti.

- **Ex4(file):** scrivere una funzione Python che prende in ingresso il nome di un **file** csv contenente le informazioni sulle amicizie e inimicizie che si creano in un gruppo di persone nel seguente formato:

  ```
  Nome1, Nome2, relazione
  ```

  dove la relazione può essere solo un valore tra `'amici'` e `'nemici'`. Dovete leggere il **file** e costruire un dizionario con chiave il nome della persona e valore la lista ordinata (usare il metodo `sort()` delle liste) dei suoi amici rimasti. Ogni volta in cui 2 persone diventano amiche dovete aggiungere ad ognuno dei due il nome dell'altro nell'elenco degli amici, se non era già presente (non ci devono essere duplicati nella lista). Se due persone diventano nemiche dovete eliminare in entrambi il nome dell'altro nell'elenco degli amici (se c'era, altrimenti non dovete fare niente).
  Ad esempio, se **file** contiene:

  ```
  Nome1, Nome2, relazione
  Paolo, Marco, amici
  Anna, Maria, amici
  Paola, Anna, amici
  Marco, Giorgio, amici
  Giorgio, Marco, nemici
  ```

  la funzione deve restituire:

  ```
  {'Marco': ['Paolo'], 'Maria': ['Anna'], 'Paolo': ['Marco'],
  'Paola': ['Anna'], 'Giorgio': [], 'Anna': ['Maria', 'Paola']}
  ```

- **Ex5(file)**: un indirizzo IP è un'etichetta numerica che identifica univocamente un dispositivo all'interno di una rete informatica. Esso è costituito da una sequenza di 4 numeri compresi tra 0 e 255 separati da un punto (es. 192.168.0.1). Tra tutti i possibili indirizzi IP, vi è un intervallo dedicato alle reti domestiche, i quali hanno formato 192.168.X.Y, dove X ed Y sono due numeri compresi tra 0 e 255. Scrivere la funzione Python che preso in ingresso il nome di un **file** di testo avente il seguente formato

  ```
  Indirizzo_IP
  Indirizzo_IP
      […]
  Indirizzo_IP
  ```

  Ritorni un dizionario contenente le seguenti chiavi: 'invalidi', 'domestici' ed 'altri', e come valori associati il numero di indirizzi letti nel file rispettivamente non validi, domestici oppure validi non domestici.

- **Ex6(file):** scrivere una funzione Python che prende in ingresso un file di testo contenente dei codici fiscali, scritti uno per riga e che possono contenere o meno spazi tra i vari campi, e restituisce la lista (nell'ordine in cui sono nel file) delle date di nascita nel formato dd/mm/aaaa (2 cifre obbligatorie per giorno e mese, 4 per anno). Si assuma che se l'anno xx nel codice fiscale è minore od uguale a 18 allora l'anno corrisponde a 20xx, altrimenti l'anno corrisponde a 19xx. Si ricorda che il formato dei codici fiscali è:

  ABC XYZ aaMgg WnnnV

  Dove ABC e XYZ sono sequenze di lettere MAIUSCOLE prese rispettivamente dal cognome e dal nome, aa denota le ultime due cifre dell'anno, M è una lettera maiuscola che specifica il mese secondo la seguente tabella riportata a lato, gg denota il giorno di nascita con la regola che se è maggiore di 40 allora il sesso è femminile e per calcolare la data corretta bisogna togliere 40. L'ultima parte indica il codice del comune (o stato estero) di nascita, composto da una lettera maiuscola e 3 cifre, mentre l'ultima lettera maiuscola è un carattere di controllo. I 4 campi possono essere separati da spazi bianchi oppure essere attaccati. Se la riga NON contiene un codice fiscale corretto dovete inserire nella lista la stringa 'Codice errato', se il codice del mese è inesistente allora inserite nella lista la stringa 'Mese errato', se il giorno è scorretto allora inserite nella lista la stringa 'Giorno errato'. Ad esempio, se il file contiene:

Lettera	Mese	Lettera	Mese	Lettera	Mese
A	gennaio	E	maggio	P	settembre
B	febbraio	H	giugno	R	ottobre
C	marzo	L	luglio	S	novembre
D	aprile	M	agosto	T	dicembre

  VXRTRR71C12H501W
  PSCTRS 21S33 P
  CVV PSX 11D55 H911T
  CVV PSX 11O55 H911T
  CVV PSX 11D79 H911T

  La vostra funzione deve restituire la lista ['12/03/1971', 'Codice errato', '15/04/2011', 'Mese errato', 'Giorno errato']