

14.09.2017

Esame di Fondamenti Informatica per Ingegneria Gestionale - A.A. 2016/2017

Durata 1h45' - Compito A

Istruzioni (leggere attentamente)

Nota importante: la mancata osservanza delle seguenti regole può comportare la perdita di informazioni necessarie alla valutazione dell'esame.

Cartella di esame e registrazione dei dati dello studente

La cartella Esame contenente il compito da svolgere si trova sul Desktop. Entrare nella cartella e, prima di iniziare il compito, eseguire il programma registrazione.pyc. Inserire i dati personali fornendo (separatamente) Numero di Matricola, Cognome e Nome. Il programma genera un file studente.txt che *non deve essere modificato manualmente*. Verificare che i dati nel file studente.txt siano corretti; in caso di errore potete rieseguire il programma registrazione.pyc.

Svolgimento degli esercizi

Leggere attentamente il testo e risolvere gli esercizi proposti.

Per ogni esercizio avete una cartella EsercN che contiene un file dal nome A_ExN.py (dove N è il numero dell'esercizio). Questo è il file che dovrete modificare con la vostra soluzione. Non create nuovi file.

Per verificare la correttezza di un esercizio **DOVETE** usare il programma TestEx.pyc (basta cliccarci sopra 2 volte) che proverà la vostra soluzione con un certo numero di casi di test. **Si noti che per la correzione verranno usati insieme di dati di test diversi.**

E' possibile consultare la documentazione ufficiale del linguaggio Python, ma **non è possibile usare libri o appunti**. In caso di problemi tecnici chiedere ai docenti o ai tecnici del laboratorio.

Esercizi

- **A_Ex1(s, c) (8 punti)** Completare la funzione Python **A_Ex1(s, c)**, che prende in ingresso una stringa s e un carattere c. La funzione deve restituire una stringa ottenuta da s eliminando ogni occorrenza di c. Ad esempio, se s fosse "amaca" e c fosse "a", la funzione dovrebbe restituire "mc". Se s è la stringa vuota la funzione deve restituire la stringa vuota.
- **A_Ex2(m) (8 punti)** Completare la funzione Python che riceve in ingresso una matrice m e restituisce una nuova matrice che rappresenta la matrice triangolare inferiore estratta da m. Se la matrice m non è quadrata, la funzione dovrà restituire la matrice vuota ([] in Python).

Esempio: Se la matrice m è $\begin{pmatrix} 0 & 1 & 2 & 3 \\ 0 & 1 & 2 & 3 \\ 0 & 1 & 2 & 3 \\ 0 & 1 & 2 & 3 \end{pmatrix}$ allora la matrice risultato sarà $\begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 2 & 0 \\ 0 & 1 & 2 & 3 \end{pmatrix}$.

Se invece la matrice m è $\begin{pmatrix} 0 & 1 & 2 & 3 \\ 0 & 1 & 2 & 3 \\ 0 & 1 & 2 & 3 \end{pmatrix}$ allora la matrice risultato sarà () ([] in Python).

- **A_Ex3(files, s) (8 punti)** Implementare la funzione **A_Ex3(files, s)** che prende in ingresso una lista di stringhe (**files**) e una stringa. Gli elementi della lista sono nomi di file testo, privi di punteggiatura. La funzione deve restituire un dizionario avente per chiavi i nomi dei file. Il valore associato a una

chiave deve invece essere il numero di occorrenze di **s** nel file corrispondente. Eventuali maiuscole devono essere ignorate, per cui ad esempio “Casa” e “casa” vanno considerate la stessa parola. Se la stringa non è contenuta in alcuno dei file, la funzione deve restituire il dizionario vuoto.

Esempio: si supponga che **files** = [“file1”, “file2”] e **s** = “panca”. Si supponga che file1 contenga il testo:

“Sopra la panca la capra campa sotto la panca la capra muore”

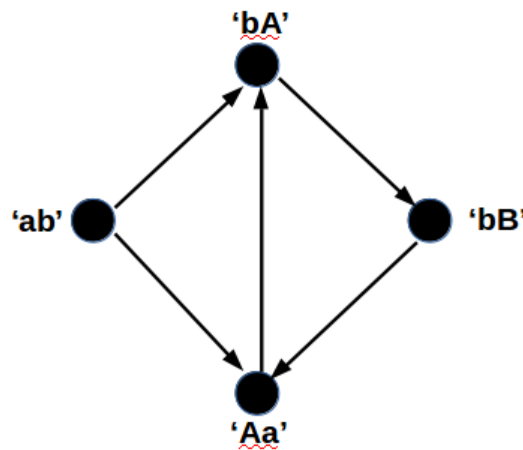
e che file2 contenga il testo:

“Sotto la panca dorme un gatto”

La funzione deve restituire la lista {“file1”: 2, “file2”: 1}. Se invece **s** = “dorme”, la funzione dovrebbe restituire {“file1”: 0, “file2”: 1}.

Nota: si ricordi che, data una stringa, il metodo `lower` restituisce una copia della stringa con caratteri minuscoli.

- **A_Ex4(g, u) (8 punti)** Implementare la funzione **A_Ex4(g, u)** che (tra gli altri parametri) riceve in ingresso un dizionario **g** che rappresenta un grafo diretto, i cui nodi sono stringhe. Il parametro **u** è un nodo del grafo. La funzione deve restituire la concatenazione in ordine alfabetico delle etichette dei nodi raggiungibili da **u**. Si assuma che ogni nodo sia sempre raggiungibile da se stesso.



Ad esempio, se il grafo fosse quello in figura, avremmo **g** = {“bA”: [“bB”], “bB”: [“Aa”], “Aa”: [“bA”], “ab”: [“bA”, “Aa”]}. Se **u** fosse il nodo “bA” la funzione dovrebbe restituire “AabAbB”. Si noti che il nodo di partenza è sempre raggiungibile.