

Fondamenti di Informatica I (12 cfu) - A.A. 2014-2015

Corsi di Laurea in Ingegneria Gestionale

Sapienza Università di Roma

Prova al calcolatore - COMPITO A

Appello del 26 Giugno 2015 - Durata 1h 45'

Note (leggere attentamente)

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

Registrazione dei dati dello studente Prima di iniziare il compito, cliccare su **Risorse del Computer**, aprire prima la cartella (di rete) che trovate nella finestra e poi la cartella **Esame**. Eseguire il programma `registrazione.pyc`, inserire i dati personali fornendo Nome, Cognome e Matricola. 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.bat`.

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) con lo scheletro della soluzione. Per provare 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.

Nota bene: il file `A_ExN.py` deve contenere SOLO la funzione `A_ExN`.

È possibile consultare la documentazione ufficiale del linguaggio Python (disponibile premendo F1 all'interno di IDLE), ma non è possibile usare libri o appunti. In caso di problemi tecnici chiedere ai docenti o ai tecnici del laboratorio.

Esercizio 1

(8 punti) Scrivere una funzione che riceve in ingresso una stringa `s` e calcoli la lunghezza della più lunga sequenza che a partire dall'inizio (posizione 0) si ripete almeno 2 volte. Ad esempio, se la stringa `s` vale `'aabbcaabb'` allora la funzione deve restituire 4 perché la sequenza `'aabb'` che parte dalla posizione 0 si ripete 2 volte e nessuna sequenza più lunga si ripete almeno 2 volte.

Esercizio 2

(8 punti) Scrivere una funzione Python che riceve in ingresso due liste **a** e **b** numeriche di uguale lunghezza NON NECESSARIAMENTE ORDINATE e restituisca una lista **c** di lunghezza doppia contenente tutti gli elementi di **a** e **b** alternati dal piu' piccolo di **a** al piu' grande di **a** e dal piu' grande di **b** al piu' piccolo di **b**. Ossia, il primo elemento di **c** deve essere uguale al piu' piccolo elemento di **a** il secondo elemento di **c** deve essere uguale al piu' grande elemento di **b** il terzo elemento di **c** deve essere uguale al seguente elemento piu' piccolo di **a** il quarto elemento di **c** deve essere uguale al seguente elemento piu' grande di **b** etc. Ad esempio, se le liste sono **a**=[1,3,2] e **b**=[5,6,4] allora **c**=[1,6,2,5,3,4]

Esercizio 3

(9 punti) Scrivere una funzione Python che prende come parametri una matrice di stringhe **m** ed un carattere **c** e restituisce l'indice della colonna in cui il carattere **c** occorre un numero dispari di volte. Se esistono due colonne in cui **c** occorre un numero di volte dispari, allora la funzione restituisce l'indice minore. Se **c** non occorre in alcuna colonna o occorre solo un numero di volte pari, la funzione restituisce -1. Ad esempio, se la matrice **m** fosse [["casa", "cielo", "bacio"], ["pappa", "luna", "orca"], ["anello", "lente", "raggio"]] ed il carattere **c** fosse 'a' allora la funzione dovrebbe restituire il valore 0, in quanto la colonna 0 contiene un numero dispari (5 in totale) di 'a'.

Esercizio 4

(7 punti) Completare la funzione Python **A_Ex4(g, v)** (contenuta nel file **A_Ex4.py**) che prende in ingresso un dizionario **g** che rappresenta un grafo *diretto* e l'etichetta **v** di un nodo e restituisce **True** se *tutti* i nodo del grafo sono raggiungibili da **v**, **False** altrimenti.

Si assuma che il grafo non contenga nodi isolati, cioè privi sia di archi entranti che uscenti. Si assuma anche che le etichette dei nodi siano stringhe. La Figura 1 chiarisce il quesito con un esempio.

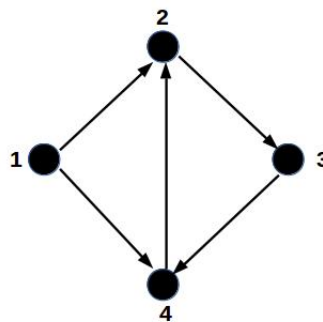


Figure 1: A titolo di esempio, se **g** rappresenta il grafo in figura, allora il metodo deve restituire **True** se **v** = '1', mentre deve restituire **False** per tutti gli altri nodi.