Programmazione Avanzata
Appello del 12/1/2022
a.a. 2021-22

Istruzioni

Svolga gli esercizi nei file indicati e dopo aver concluso tutti gli esercizi comunichi alla docente di voler consegnare. Dopo aver ricevuto l'ok dalla docente mostri uno alla volta i file degli esercizi alla docente. Da quel momento non potra` piu` modificare il codice dell'esercizio e dovra` salvare tutti i file .py degli esercizi in una cartella zippata il cui nome deve essere formato dal suo cognome e nome separati da un '_' (ad esempio, rossi_mario). Invii la cartella zippata a adebonis@unisa.it con oggetto "Soluzione esercizi appello dicembre".

Durante lo svolgimento della prova, eccezion fatta per il collegamento al meeting di Zoom, è vietato l'uso di internet (navigazione tramite browser, posta, ssh, ftp, accesso remoto al PC, condivisione dello schermo e di qualsiasi cartella, ecc.).
Teams e Zoom devono essere ridotti ad icona.

NON MODIFICHI IL CODICE GIA` SCRITTO NEI FILE FORNITI DALLA DOCENTE (0 punti se cio` viene fatto)

1. Scrivere nella classe C, fornita nel file esercizio1.py, un metodo di classe **aggiungiProprieta** che per ciascuna variabile di classe di C di tipo str, crea una property con lo stesso nome della variabile. Il setter della property deve effettuare l'assegnamento solo se il valore da assegnare è una stringa. In caso contrario deve lanciare TypeError con argomento la stringa "Non e` possibile assegnare {} alla variabile {}" dove al posto delle parentesi graffe devono comparire il valore e il nome passati al setter.

2. Scrivere nel file esercizio2.py un decorator factory dFact che restituisce un decoratore di funzione che "trasforma" la funzione f in un generatore che all'i-esima invocazione di next restituisce il valore ottenuto invocando f con gli argomenti ottenuti sommando L[i] a tutti gli argomenti con cui la funzione f è invocata originariamente. Se la suddetta somma causa un'eccezione TypeError allora il generatore smette di restituire valori.

3. Scrivere nel file esercizio3.py una classe C per cui accade che ogni volta che si aggiunge una variabile di istanza ad una delle istanze di C in realta` la variabile viene aggiunta alla classe come variabile di classe.
   Versione con Bonus: modificare il codice in modo tale che le istanze abbiano al piu` due variabili di istanza: varA e varB (non viene creato __dict__) e non deve essere possibile aggiungere altre variabili di istanza oltre a queste due. Se il programma avesse bisogno di aggiungere altre variabili oltre a quelle sopra indicate, queste altre variabili verrebbero create come variabili di classe e non di istanza.

Programmazione Avanzata
Appello del 12/1/2022
a.a. 2021-22

4. Scrivere la funzione **stampa** all'interno del file esercizio4.py. Se la funzione ha bisogno di invocare altre procedure, fornire anche queste ultime. La funzione **stampa** prende in input una parola **P**, una collezione iterabile di nomi di file **listaDiFile,** e il parametro **concorrenza**. Facendo uso di **multiprocessing.JoinableQueue**, la funzione **stampa** deve stampare per ciascuno dei file di **listaDiFile** "La parola {} appare nel file {} in posizione {}.", dove al posto delle parentesi devono comparire **P**, **il nome del file** e **la posizione in cui P appare per la prima volta nel file**. Se **P** non appare nel file allora **stampa** deve stampare "La parola {} non appare nel file {}." dove al posto delle parentesi graffe devono comparire, rispettivamente, **P** e **il nome del file**. La ricerca della parola deve essere effettuata con un processo separato per ogni file di **listaDiFile** e **le stampe devono essere effettuate nell'ordine in cui terminano i processi e quando sono stati elaborati tutti i file. Il callable usato da stampa deve prendere in input la collezione dei nomi dei file e la parola.**

   Suggerimento: per restituire la posizione in cui è stata trovata la parola invocate tell() sul file object (in questo modo si ottiene la posizione in cui finisce la parola). Invocate il metodo tell() fuori dal ciclo in cui effettuate la ricerca.