ZOO-2: classi e oggetti per gestire un collezione di animali con una GUI

Alessandro Bocci, Susanna Pelagatti

Terzo assegnamento in itinere 622AA
AA 2024-25

Indice

1 Introduzione

Programmazione e Analisi Dati (622AA) per la parte Python prevede lo svolgimento di tre assegnamenti in itinere (uno Scratch e due Python) che esonerano dallo svolgimento del progetto finale. Questo documento descrive il primo assegnamento Python che prevede la realizzazione di un insieme di funzioni Python.

Il software viene sviluppato e documentato utilizzando gli strumenti, le tecniche e le convenzioni presentati durante il corso.

1.1 Materiale in linea

Tutto il materiale relativo al corso può essere reperito dal Moodle ufficiale del corso. Eventuali chiarimenti possono essere richiesti ai docenti per posta elettronica.

1.2 Struttura degli assegnamenti, bonus, tempi di consegna e prova orale

I tre assegnamenti possono essere svolti individualmente o in gruppi di 2 studenti e possono essere consegnati entro il 31 Gennaio 2025. Gli studenti che

consegnano una realizzazione sufficiente entro la data stabilita dai docenti per ciascun assegnamento avranno un incremento di 2 punti (di Bonus) nella valutazione complessiva di ciascun assegnamento. La valutazione complessiva viene data dalla media delle valutazioni dei tre assegnamenti e costituisce la base per la prova orale. La prova orale sarà composta di due parti. La prima parte sarà una discussione sugli assegnamenti consegnati e tenderà a stabilire se lo studente è realmente l'autore di quanto consegnato (in caso di dubbi la valutazione verrà opportunamente aggiustata) in particolare verrà chiesto di leggere e modificare il codice e di spiegare quanto usato (in particolare costrutti e moduli non facenti parte del programma del corso). La seconda parte verterà su tutto il programma del corso. In particolare, l'orale comprenderà:

- una discussione delle scelte implementative
- l'impostazione e la scrittura di semplici programmi Python (sequenziali e GUI) di difficoltà medio bassa rispetto a quelli visti nelle esercitazioni in classe
- domande su tutto il programma presentato durante il corso.

Il voto finale comprenderà la valutazione effettiva degli assegnamenti (emendata in caso di dubbi) e la prova orale.

1.3 Consegna degli assegnamenti

La consegna degli assegnamenti avviene *esclusivamente* attraverso il Moodle del corso, seguendo le istriuzioni nei file `README` di ciascun assegnamento..

1.4 Valutazione dell'assegnamento

All'assegnamento viene assegnata una fascia di valutazione da 0 a 30 che tiene conto dei seguenti fattori:

- motivazioni, originalità ed economicità delle scelte implementative.
- strutturazione del codice (fattorizzazione del codice in funzioni, uso di strutture dati adeguate etc).
- efficienza e robustezza (numero di operazioni eseguite, fallimenti in caso di input inadeguati etc).
- aderenza alle specifiche.
- qualità del codice Python e dei commenti.
- usabilità della GUI.

Tutti gli assegnamenti verranno confrontati automaticamente per verificare situazioni di plagio. Nel caso di elaborati uguali verranno presi provvedimenti per tutti i gruppi coinvolti.

2 L'assegnamento `ZOO-2`

L'assegnamento prevede la realizzazione di uno zoo simile (ma non uguale!) a quello del secondo assegnamento, attraverso la definizione di classi e di una interfaccia grafica per attivare alcuni metodi dello zoo e per visualizzare gli animali in esso ospitati.

L'assegnamento prevede la definizioni di una gerarchia di classi di tre livelli per rappresentare gli animali e una classe per rappresentare lo zoo.

2.1 Gli Animali

Gli animali devono essere rappresentati da una gerarchia di classi

$$\text{Animale} > \textit{Classe\_Animale} > \textit{Specie\_Animale}$$

Dove Classe_Animale rappresenta la classe dell'animale (uno fra "Mammifero", "Uccello" o "Pesce") e Specie_Animale rappresenta il nome delle Specie a cui appartiene l'animale, per esempio Leone, Giraffa, etc. Il simbolo > indica la relazione di superclasse > sottoclasse. La lista delle classi di specie da implementare è Leone, Giraffa, Ippopotamo, Pinguino, Gufo, PescePagliaccio.
Per esempio, per un leone la gerarchia è:

$$\text{Animale} > \text{Mammifero} > \text{Leone}$$

Dove la classe Leone estenderà Mammifero che estenderà Animale. Nel caso di una specie di mammifero diversa, basterà estendere Mammifero per aggiungere la specie alla gerarchia.

Lo stato e i metodi da implementare per ogni classe sono specificati nel file `animali.py`. Ogni classe che modifica lo stato dovrà controllare il tipo e il valore delle variabili in input e lanciare opportune eccezioni.

2.2 Lo Zoo

La classe che rappresenta lo zoo deve gestire la collezione di animali, implementando almeno le specifiche del file `zoo.py`.

Come nell'assegnamento precedente, lo zoo è suddiviso in zone rappresentate da una lettera maiuscola e una cifra decimale (es: A1, B7, Z0).

Lo stato dello zoo consiste in un dizionario che ha come chiave il nome di un animale (che deve essere univoco) e cove valore una coppia (*animale*, *zona*), dove *animale* è un oggetto di una classe Animale o di una sua sottoclasse, mentre *zona* è una stringa formattata come spiegato sopra.

2.3 La GUI

Deve essere realizzata un'interfaccia grafica usando il modulo `tkinter` che permetta la gestione dello zoo senza interagire con o dover leggere la linea di comando.

Dall'interfaccia grafica deve essere possibile visualizzare gli animali (bastano il nome e la specie) dello zoo divisi per zone. Inoltre, usando l'interfaccia grafica deve essere possibile:

- inserire un nuovo animale nello zoo,

- modificare la zona di un animale,
- mostrare la stringa rappresentante un animale dato il nome,
- ottenere la lista dei nomi degli animali appartenenti a una specie,
- ottenere la lista degli animali che vivono in un determinato ambiente,
- ottenere il numero di animali data una classe,
- caricare lo zoo da file,
- salvare lo zoo su file,
- uscire dall'applicazione.

Per operazioni non intuitive per un utente della GUI, prevedere un meccanismo di aiuto. Per esempio, l'inserimento dei dati di un nuovo animale potrebbe richiedere un formato specifico; cliccare su bottone avente come testo "?" potrebbe apparire una messagebox informativa con il formato da usare.

Utilizzare il file `gui.py` per implementare l'interfaccia grafica, con gli `import` inseriti è possibile sfruttare le classi definite nel file `zoo.py` (devono essere conenuti nella stessa cartella).

L'aspetto dell'interfaccia grafica è libero. La cosa importante è che siano implementate tutte le operazioni elencate sopra e che un utente che usa solo la GUI sia in grado di utilizzare il programma facilmente. Un uso avanzato di `tkinter` per rendere l'interfaccia più gradavole e usabile, sarà considerato al pari di aggiungere funzionalità extra al codice della classe zoo.

Il formato della rappresentazione dello zoo su file è libero. Un consiglio è di utilizzarne uno per il quale sia più facile caricare il file piuttosto che il file sia facilmente leggibile da un essere umano.

Gestire accuratamente le eccezioni e i loro messaggi per restituire feedback all'utente.

2.4 Istruzioni

Unitamente a questo file vengono forniti i file Python contenenti l'intestazione delle funzioni da realizzare (`zoo.py` e `animali.py`), un file con solo gli import di base in cui dovrà essere implementata la GUI (`gui.py`), i test che sevono essere superati per consegnare il codice (`main.py` e `testMy.py`) e un file di istruzioni (`README`). I test del fle `main.py` testeranno le classi e i metodi implementati in `animali.py` e `zoo.py`, non l'interfaccia grafica.

Consigliamo di leggere attentamente le istruzioni e analizzare il codice fornito per capire come strutturare le funzioni prima di iniziare a progettare e implementare. Ricordiamo l'importanza di analizzare i vari casi prima di iniziare a scrivere codice e di effettuare test incrementali sul codice durante lo sviluppo.

Possono essere realizzate funzionalità in più rispetto a quelle richieste (ad esempio altre funzioni).

Le parti opzionali devono essere corredate da test appropriati e documentate da commenti chiari o (in caso sia necessario) da un breve documento descrittivo che puo' essere consegnato insieme al codice e che spiega le motivazioni e la struttura di quanto realizzato.