**Emanuele Scopelliti esame del 6/12/2024**

**Traccia:** Per agire come un Hacker bisogna capire come pensare fuori dagli schemi. L'esercizio di oggi ha lo scopo di allenare l'osservazione critica. Dato il codice si richiede allo studente di:

● Capire cosa fa il programma senza eseguirlo. Esercizio Progetto

● Individuare nel codice sorgente le casistiche non standard che il programma non gestisce (esempio, comportamenti potenziali che non sono stati contemplati).

● Individuare eventuali errori di sintassi / logici.

● Proporre una soluzione per ognuno di essi.

```
1 import datetime
2 def assistente_virtuale(comando):
3         if comando == "Qual è la data di oggi?":
4                 oggi = datetime.datetoday()
5                 risposta = "La data di oggi è " + oggi.strftime("%d/%m/%Y")
6         elif comando == "Che ore sono?":
7                 ora_attuale = datetime.datetime.now().time()
8                 risposta = "L'ora attuale è " + ora_attuale.strftime("%H:%M")
9         elif comando == "Come ti chiami?":
10                risposta = "Mi chiamo Assistente Virtuale"
11        else:
12                risposta = "Non ho capito la tua domanda."
13        return risposta
14        while True
15                comando_utente = input("Cosa vuoi sapere? ")
16                if comando_utente.lower() == "esci":
17                        print("Arrivederci!")
18                        break
19                else:
20                        print(assistente_virtuale(comando_utente))
21
```

**1)Esercizio Capire cosa fa il programma.**

Il programma è in una forma elementare di assistente virtuale, fornisce delle risposte in base ad alcune domande prestabilite che possiamo scegliere.

Come prima fase ci stampa la frase:

"Cosa vuoi sapere?"

In questa prima richiesta abbiamo la scelta tra esci oppure inserisci un comando prestabilito

I Comandi sono:

"Qual è la data di oggi?"

"Che ore sono?"

"Come ti chiami?"

Nel caso in cui in questa fase non scegli tra nessuna di queste opzioni o inserisci caratteri o numeri errati ci fornisce la risposta:

“Non ho capito la tua domanda”

E torna al menù principale in Loop finché non inseriamo un comando corretto.

**Analizziamo alcuni comandi della libreria datetime utilizzati:**

---

import datetime: importiamo la libreria relativa alle funzioni di Ora e Data

datetime.date.today(): restituisce la data corrente

strftime(): Converte il formato della data del datetime in una stringa formattata a nostro piacimento nello specifico abbiamo voluto che inserisse in ordine %d/%m/%Y  ovvero d= DAY m=Month Y=Year

datetime.datetime.now().time(): restituisce l’orario corrente relativo al fuso orario che è impostato il PC

comando_utente.lower(): converte in qualunque modo si scriva EsCi in caratteri in minuscolo “esci”

**Analizziamo il codice in blocco:**

---

Il codice io lo divido in tre blocchi

Il primo blocco l’aggiunta della libreria datetime

```
1 import datetime
```

il secondo blocco corrisponde alla creazione della funzione di nome “assistente_virtuale” che utilizza una variabile di nome “comando”

In questo blocco troviamo le varie scelte che può assumere la variabile “comando”

```
2 def assistente_virtuale(comando):
3         if comando == "Qual è la data di oggi?":
4                 oggi = datetime.datetoday()
5                 risposta = "La data di oggi è " + oggi.strftime("%d/%m/%Y")
6         elif comando == "Che ore sono?":
7                 ora_attuale = datetime.datetime.now().time()
8                 risposta = "L'ora attuale è " + ora_attuale.strftime("%H:%M")
9         elif comando == "Come ti chiami?":
0                 risposta = "Mi chiamo Assistente Virtuale"
1         else:
2                 risposta = "Non ho capito la tua domanda."
3         return risposta
```

Il terzo blocco che inizia dal while True, anche se è scritto alla fine del programma, corrisponde visivamente per l’utente utilizzatore alla parte iniziale, il programma andrà a leggere esattamente come primi comandi dal while True in giù

```
14        while True
15                comando_utente = input("Cosa vuoi sapere? ")
16                if comando_utente.lower() == "esci":
17                        print("Arrivederci!")
18                        break
19                else:
20                        print(assistente_virtuale(comando_utente))
```

E solo dopo la chiamata alla funzione “assistente_virtuale” ritornerà in cima ed entrerà appunto dentro la funzione “assistente_virtuale”

**2) Individuare nel codice sorgente le casistiche non standard che il programma non gestisce (esempio, comportamenti potenziali che non sono stati contemplati).**

---

- il programma entra in loop infinito fintanto che si inserisce una risposta non corretta
- il programma accetta comandi solo se sono scritti in maniera identica a come indicato ed è anche Case Sensitive, non accetta mancanze di caratteri o errori di nessun genere nel comando
- accetta tutti i tipi di caratteri e numeri
- il "datetime.datetime.now().time()", restituisce l'orario corrente relativo al fuso orario che è impostato il PC e non un fuso orario diverso o magari quello in cui è locato il PC, immaginiamo di usare il PC in Italia ma il fuso orario del PC è impostato su NewYork ci restituirà l'orario di NewYork

**3) Individuare eventuali errori di sintassi / logici.**

---

**Errore di sintassi:**

datetime.datetoday() manca il "."

Versione Corretta

datetime.date.today()

Errore nel while True manca il simbolo dei ":" ed inoltre anche l'indentazione è errata:

white True

corretto

while True:

while True la correzione di indentazione consiste nel posizionamento della riga di codice nel punto corretto. In questa casistica andava inserito senza usare nessun TAB ovvero allineandolo al margine sinistro, perché è nel blocco più esterno del programma.

**Errori logici:**

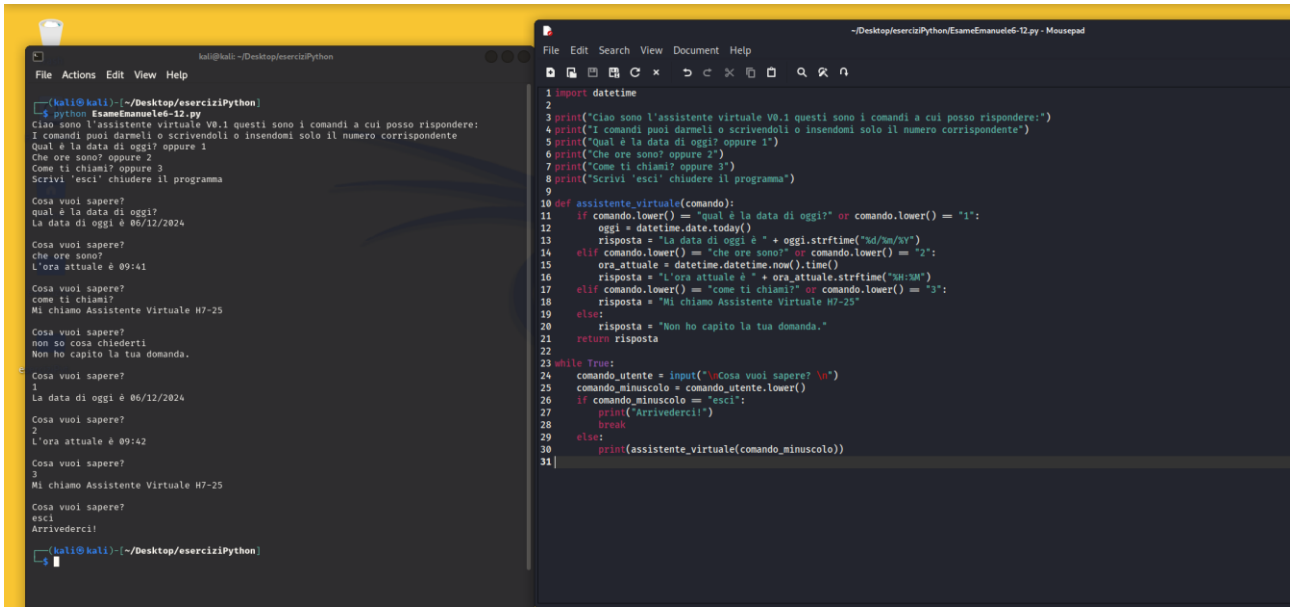
A) Il programma è in costante loop sia se si sceglie una domanda corretta che inserendo domande errate, è dovuto intrinsecamente all'utilizzo del While True perché la condizione del programma per entrare al suo interno sarà sempre vera, esce solo e soltanto inserendo il comando "esci", può essere sia considerato un errore in base all'utente utilizzatore che anche essere una condizione corretta, in questa casistica il programma rimane in costante ascolto di un input dall'utente.
B) I comandi sono sensibili al Case Sensitive, ovvero vanno scritti esattamente per come sono richiesti rispettando maiuscole e minuscole

C) L'utente finale non conosce i comandi all'interno del programma vanno stampati all'apertura del programma
D) Una semplificazione è quella di assegnare un valore numerico ad ogni comando in modo da inserire solo il valore numerico per ottenere la risposta senza dover inserire tutto il comando per intero, risparmiando errori di battitura e tempo
E) Il programma viene visualizzato con le domande e le risposte nelle stesse righe, quindi le domande e le risposte vanno inserite su righe diverse per essere più leggibili.
F) Il Fuso orario non è modificabile

**4) Proporre una soluzione per ognuno di essi.**

---

Questo è il programma che ho riscritto seguendo gli errori segnalati e con di fianco lo screen in funzione con tutte le possibili scelte




A) Nel caso in cui consideriamo il punto A come un errore per correggerlo basta riscrivere il programma senza usare il while True e possiamo anche considerare di non inserire l'assistente virtuale come una Funzione ma semplicemente come un Menù interattivo.

B) Riguardo l'errore logico del Case Sensitive ho risolto con due metodi il problema inserendo il comando .lower() a tutti i comandi dati dall'utente in modo da poter inserire il comando anche non rispettando il case sensitive.

C) Per poter dare la possibilità all'utente finale di saper i comandi all'interno del programma, il software una volta avviato li stampa nelle prime righe informando l'utente delle scelte possibili

D) Per risparmiare tempo ed errori di battitura ho inserito la possibilità di digitare un valore numerico in base al comando voluto, quindi basta digitare il valore numerico corrispondente. Questa opzione è stata aggiunta inserendo un semplice OR ad ogni scelta di comando

E) Per rendere il programma più leggibile e presentabile all'utente finale ho inserito alcuni a capo attraverso lo \n

F) Va implementata una funzione che io non ho inserito per motivi tempistici in cui si chiede le date e l'ora in base ad un fuso orario scelto dall'utente

D) Tutti gli errori di sintassi e di indentazione sono stati corretti