REPORT S2/L5

-Indice:

---

1) introduzione :

L’esercizio di oggi è incentrato sull’analisi critica del codice. L'obiettivo è esaminare il programma fornito senza eseguirlo, individuando eventuali errori e possibili miglioramenti. Questo tipo di esercizio permette di affinare le capacità di osservazione e problem-solving, fondamentali nel mondo della programmazione e della cybersecurity.

2) Motivo dell'esercizio:

Questo esercizio insegna a ragionare come un hacker: anziché eseguire il codice immediatamente, si deve analizzarlo a mente, individuando errori di sintassi, bug logici e casi limite non gestiti.

3) Scopo dell' esercizio :

L’obiettivo finale è correggere il codice per garantire un funzionamento corretto e sicuro, evitando errori e migliorando l’interazione con l’utente. Con queste correzioni, il programma diventa più stabile, leggibile e robusto.

4) descrizione dell'attività che andremo a svolgere :

Il codice fornito rappresenta un assistente virtuale in grado di rispondere a tre domande specifiche:

- Qual è la data di oggi?

- Che ore sono?

- Come ti chiami?

Il programma utilizza il modulo **datetime** per recuperare la data e l'ora attuale e restituisce una risposta testuale.

Quello che andremo a fare è un **Code Review**, analisi degli errori, correzioni e creazione del codice funzionante.

5) Esposizione del codice python fornito :

```
import datetime

def assistente_virtuale(comando):
    if comando == "Qual è la data di oggi?":
        oggi = datetime.datetoday()
        risposta = "La data di oggi è " + oggi.strftime("%d/%m/%Y")
    elif comando == "Che ore sono?":
        ora_attuale = datetime.datetime.now().time()
        risposta = "L'ora attuale è " + ora_attuale.strftime("%H:%M")
    elif comando == "Come ti chiami?":
```

```
        risposta = "Mi chiamo Assistente Virtuale"
    else:
        risposta = "Non ho capito la tua domanda."
    return risposta
while True
comando_utente = input("Cosa vuoi sapere? ")
if comando_utente.lower() == "esci":
    print("Arrivederci!")
    break
else:
    print(assistente_virtuale(comando_utente))
```

6) Esposizione degli errori nel codice :

Facendo un' analisi del codice appaiono sia errori logici che di sintassi, che elencherò qui sotto:

- Errori di sintassi :

1) datetime.datetoday() scritto così non funziona, la sintassi corretta è la seguente: **datetime.date.today()**

2) while true scritto così non funziona perchè mancano i due punti --> " : " in quanto python richiede quei due punti per avviare il blocco, la scrittura corretta è **while true:**

- Errori logici:

nella riga con il codice **comando_utente = input("Cosa vuoi sapere? ")** se non si

inserisce un "if not" nella stringa sotto, è probabile che se l'utente scriva qualcosa di diverso o di sbagliato, l'assistente non va in crash, ma potrebbe accettare input vuoti e dare risposte errate. Per risolvere si potrebbe utilizzare il seguente comando:

```
comando_utente = input("Cosa vuoi sapere? ")

if not comando_utente:
    print("Per favore, inserisci un comando valido.")
    continue
```

continue ---> Evita che il programma passi un comando vuoto alla funzione, facendo ripartire il loop corretto.

7) Codice corretto

```
import datetime

def assistente_virtuale(comando):
    if comando == "Qual è la data di oggi?":
        oggi = datetime.date.today()
        risposta = "La data di oggi è " + oggi.strftime("%d/%m/%Y")
    elif comando == "Che ore sono?":
        ora_attuale = datetime.datetime.now().time()
        risposta = "L'ora attuale è " + ora_attuale.strftime("%H:%M")
    elif comando == "Come ti chiami?":
```

```
        risposta = "Mi chiamo Assistente Virtuale"
    else:
        risposta = "Non ho capito la tua domanda. Prova con: 'Qual è la data di oggi?', 'Che ore sono?', 'Come ti chiami?'."

    return risposta

while True:
    comando_utente = input("Cosa vuoi sapere? ")

    if not comando_utente:
        print("Per favore, inserisci un comando valido.")
        continue

    if comando_utente.lower() == "esci":
        print("Arrivederci!")
        break
    else:
        print(assistente_virtuale(comando_utente))
```

8) Conclusione

Questo esercizio mi ha aiutato a capire quanto sia importante controllare il codice prima di eseguirlo. Analizzandolo riga per riga, ho potuto individuare errori di sintassi e logica che avrebbero compromesso il corretto funzionamento del programma.

Ora il codice è più chiaro, funziona meglio e gestisce meglio gli errori. Inoltre, ho capito quanto sia utile dare messaggi più precisi all'utente, così da rendere l'interazione più intuitiva. In generale, questa esperienza mi ha fatto vedere quanto sia importante la fase di revisione per scrivere codice più pulito ed efficiente.

Luca Tavani.