Indice

1. 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:

1. Capire cosa fa il programma senza eseguirlo.
2. Individuare nel codice sorgente le casistiche non standard che il programma non gestisce (esempio, comportamenti potenziali che non sono stati contemplati).
3. Individuare eventuali errori di sintassi / logici.
4. 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))
```

2. Soluzione

2.1 Funzionalità del programma

Il programma è scritto in *Python*, come si può osservare dall'uso delle *keyword* tipiche del linguaggio e dalla tipizzazione dinamica, che consente di dichiarare variabili senza specificarne esplicitamente il tipo. Ha lo scopo di implementare un assistente virtuale testuale che risponde a tre possibili domande:

- "Qual è la data di oggi?"
- "Che ore sono?"
- "Come ti chiami?"

Continua a chiedere input all'utente finché non viene scritto "esci".

2.2 Errori di sintassi

Un errore di sintassi si verifica quando il codice non segue le regole grammaticali del linguaggio di programmazione utilizzato, in questo caso *Python*. Nel codice fornito è possibile individuare 2 errori di sintassi che non consentono al programma di essere eseguito correttamente:

- **Riga 4.**
  Nel codice viene chiamato *datetime.datetoday()*, ma il metodo corretto è *datetime.date.today()* (manca il punto tra *date* e *today()*).

- **Riga 14.**
  Il modo in cui è stato impostato il ciclo *while* non è corretto, poiché dopo la condizione *True* vanno inseriti i due punti. Questo permette al blocco indentato successivo di essere eseguito correttamente.

2.3 Errori logici

Un errore logico si verifica quando il programma funziona tecnicamente, ma non fa quello che dovrebbe fare.

Nel codice è possibile individuare un errore logico legato alla gestione dell'orario:

- **Riga 7.**
  L'uso di *.time()* dopo *datetime.now()* è superfluo e può rendere il codice meno leggibile e leggermente più complesso senza benefici pratici.

Un ulteriore errore logico riguarda la rigidità del programma nel riconoscere i comandi inseriti dall'utente: esso accetta solo stringhe esattamente corrispondenti a quelle previste nei controlli *if*. Qualsiasi variazione, anche minima, come ad esempio *"Quale è la data di oggi?"* invece di *"Qual è la data di oggi?"*, viene interpretata come input non valido.

Una possibile soluzione consiste nell'uso del metodo *.lower()* per rendere il confronto case-insensitive, e nell'adozione di controlli più flessibili, ad esempio introducendo un dizionario di comandi o utilizzando input numerici per rappresentare le diverse azioni disponibili (es. 1 = data, 2 = ora, 3 = nome), rendendo il programma più flessibile e user-friendly.

3. Alternative e implementazioni

È possibile implementare ulteriormente il codice per migliorare la user experience. Infatti, sarebbe opportuno inserire un messaggio di benvenuto, un menu per informare l'utente sulle funzioni che può svolgere il programma, semplificare la scelta dell'azione utilizzando dei numeri anziché stringhe lunghe.

Inoltre, è possibile rendere il codice più modulare e scalabile aggiungendo un dizionario a cui collegare le azioni. Tale implementazione consente di associare ad ogni valore inserito dall'utente una determinata risposta e permette, anche, di aggiungere ulteriori azioni più facilmente.

È possibile snellire ulteriormente il codice importando dalla libreria *datetime* solo i moduli utili (*date* e *datetime*) così da rendere il codice più efficiente e pulito.

```
from datetime import datetime, date

# Funzione per mostrare il menu all'utente
def mostra_menu():
    print("\nCosa vuoi sapere? (Inserisci il numero della domanda corrispondente)\n\
1 - Qual è la data di oggi?\n\
2 - Che ore sono?\n\
3 - Come ti chiami?\n\
4 - Esci dal programma\n")

# Funzione per rispondere alle domande basate sul prompt dell'utente
def assistente_virtuale(comando):
    # Dizionario delle risposte
    azioni = {
        "1": f"La data di oggi è {date.today().strftime('%d/%m/%Y')}",
        "2": f"L'ora attuale è {datetime.now().strftime('%H:%M')}",
        "3": "Mi chiamo Assistente Virtuale"
    }

    """
    Restituisce la risposta corrispondente, oppure un messaggio di errore.
    Il metodo .get consente di accedere in modo sicuro al dizionario 'azioni'
    verificando la presenza della chiave, altrimenti restituisce errore
    """
    return azioni.get(comando, "Input non valido. Per favore inserisci numeri da 1 a 4 per selezionare un'opzione.")

print("Benvenuto!")

while True:
    # Mostra il menu
    mostra_menu()

    # Ottieni il comando dell'utente e pulisce eventuali whitespace
    comando_utente = input().strip()

    # Controlla se l'utente vuole uscire
    if comando_utente == "4":
        print("Arrivederci!")
        break  # Esce dal ciclo while

    # Altrimenti risponde in base al comando
    else:
        print(assistente_virtuale(comando_utente))
```

Esempio di output

```
Benvenuto!

Cosa vuoi sapere? (Inserisci il numero della domanda corrispondente)
1 - Qual è la data di oggi?
2 - Che ore sono?
3 - Come ti chiami?
4 - Esci dal programma

1
La data di oggi è 11/04/2025

Cosa vuoi sapere? (Inserisci il numero della domanda corrispondente)
1 - Qual è la data di oggi?
2 - Che ore sono?
3 - Come ti chiami?
4 - Esci dal programma

2
L'ora attuale è 14:00

Cosa vuoi sapere? (Inserisci il numero della domanda corrispondente)
1 - Qual è la data di oggi?
2 - Che ore sono?
3 - Come ti chiami?
4 - Esci dal programma

3
Mi chiamo Assistente Virtuale

Cosa vuoi sapere? (Inserisci il numero della domanda corrispondente)
1 - Qual è la data di oggi?
2 - Che ore sono?
3 - Come ti chiami?
4 - Esci dal programma

5
Input non valido. Per favore inserisci numeri da 1 a 4 per selezionare un'opzione.

Cosa vuoi sapere? (Inserisci il numero della domanda corrispondente)
1 - Qual è la data di oggi?
2 - Che ore sono?
3 - Come ti chiami?
4 - Esci dal programma

4
Arrivederci!
```

4. Conclusione

L'analisi del codice ha permesso di identificare errori sintattici e logici che ne impedivano il corretto funzionamento e limitavano la flessibilità nell'interazione con l'utente. Con le modifiche proposte, il programma diventa più robusto, intuitivo e facilmente estendibile.