

Intestazione: YURI PEDRANA

Introduzione:

Nel laboratorio di oggi ci è stato chiesto di eseguire il seguente esercizio:

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
- Individuare nel codice sorgente le casistiche non standard che il programma non gestisce (esempio, c omportamenti potenziali che non sono stati contemplati).
- Individuare eventuali errori di sintassi / logici.
- Proporre una soluzione per ognuno di essi.

Iniziamo...

Come primo step per individuare eventuali errori, ho riscritto il codice senza eseguirlo, per verificarne visivamente la correttezza.

Introduzione generale:

Ho preparato due slide per mostrare in modo generale gli errori riscontrati e le loro correzioni. Queste slide forniscono una panoramica dei problemi e delle soluzioni applicate.

```
import datetime
def assistente_virtuale(comando):
    if comando == "Qual è la data di oggi?":
        oggi = datetime.datetime.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."
        return risposta
while True:
    comando_utente = input("Cosa vuoi sapere?")
    if comando_utente.lower() == "esci":
        print("Arrivederci!")
        break
    else:
        print(assistente_virtuale(comando_utente))
```

```
import datetime
def assistente_virtuale(comando):
    if comando == "Qual è la data di oggi?":
        oggi = datetime.datetime.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"
    return risposta
while True:
    comando_utente = input("Cosa vuoi sapere?")
    if comando_utente.lower() == "esci":
        print("Arrivederci!")
    else: print(assistente_virtuale(comando_utente))
```

Analisi speicifica:

Adesso entriamo un po' più nel dettaglio grazie alla seguente tabella, che indica gli errori riscontrati e le relative correzioni.

ERRATO	CORRETTO
<pre>oggi = datetime.datetoday()</pre>	<pre>oggi = datetime.datetime.today()</pre>
<pre>while True</pre>	<pre>while True:</pre>
<pre>print("Arrivederci!") break else: print(assistente_virtuale(comando_utente))</pre>	<pre>print("Arrivederci!") else: print(assistente_virtuale(comando_utente))</pre>

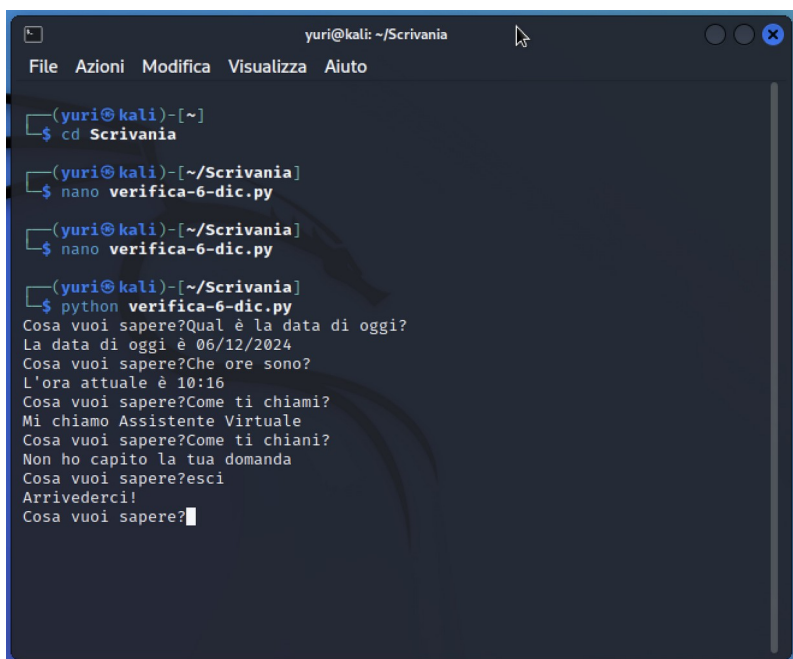
Grazie alla precedente tabella, sono stati individuati i tre errori principali in Python, ossia i più comuni:

- **Errore di sintassi:** si verifica quando il codice non è scritto correttamente. Come si può vedere, nel primo errore è stata dimenticata una parte del codice.
- **Errore di logica:** avviene quando la struttura del codice è corretta, ma la logica non produce il risultato desiderato. Nel secondo errore, manca l'uso dei due punti : necessari per avviare il blocco di codice che segue l'istruzione while. Senza i due punti, il ciclo non è valido e Python segnala un errore di sintassi, impedendo l'esecuzione delle azioni successive.
- **Errore di runtime:** si verifica durante l'esecuzione del programma e può causare un'interruzione. Nell'ultimo errore, è presente un break in più che non dovrebbe essere lì, come mostrato nella versione corretta finale. Questo perché l'utente può semplicemente digitare ESCI per uscire dal ciclo.

Conclusione:

Una volta corretti gli errori, possiamo eseguire il codice e, come risultato, vediamo che il programma restituisce le domande con le risposte corrette.

Così facendo, il lavoro è stato completato correttamente



```
yuri@kali: ~/Scrivania
File Azioni Modifica Visualizza Aiuto

(yuri@kali)-[~]
$ cd Scrivania

(yuri@kali)-[~/Scrivania]
$ nano verifica-6-dic.py

(yuri@kali)-[~/Scrivania]
$ nano verifica-6-dic.py

(yuri@kali)-[~/Scrivania]
$ python verifica-6-dic.py
Cosa vuoi sapere?Qual è la data di oggi?
La data di oggi è 06/12/2024
Cosa vuoi sapere?Che ore sono?
L'ora attuale è 10:16
Cosa vuoi sapere?Come ti chiami?
Mi chiamo Assistente Virtuale
Cosa vuoi sapere?Come ti chiami?
Non ho capito la tua domanda
Cosa vuoi sapere?esci
Arrivederci!
Cosa vuoi sapere?
```