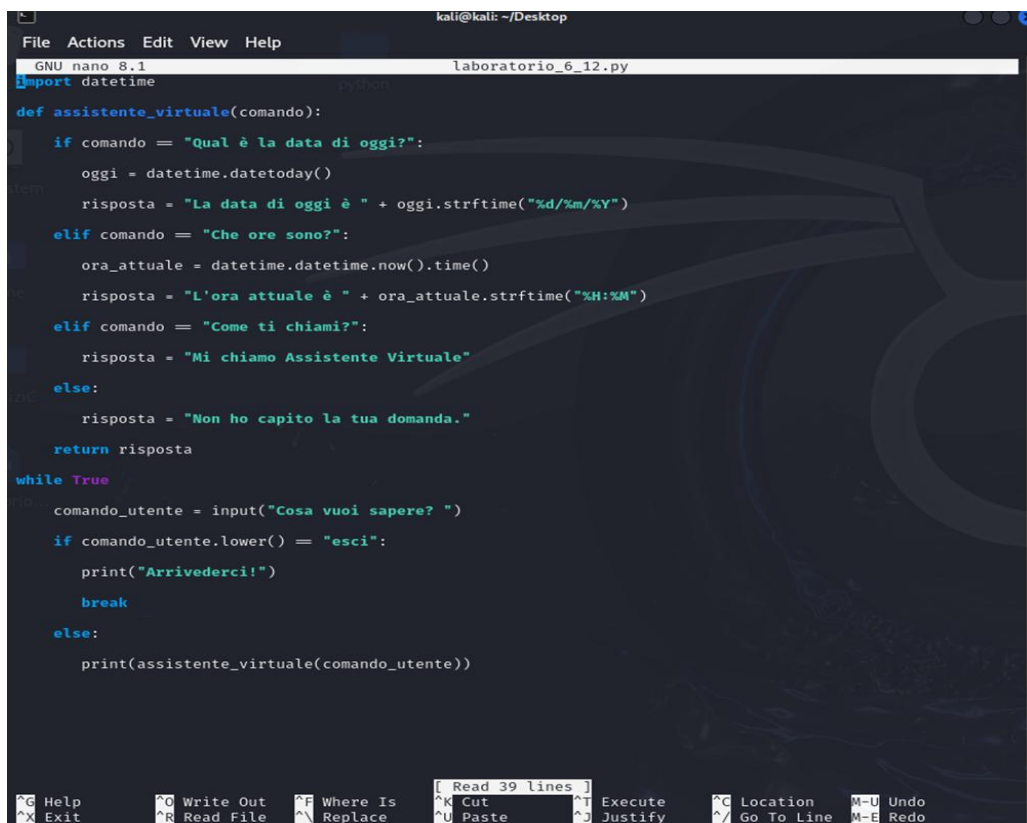


# Laboratorio 6 Dicembre 2024 S2-L5

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.

Nel laboratorio odierno dobbiamo osservare ed analizzare il seguente codice:



```
File Actions Edit View Help
GNU nano 8.1 laboratorio_6_12.py
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))

[ Read 39 lines ]
^G Help      ^O Write Out  ^F Where Is   ^K Cut        ^T Execute
^X Exit      ^R Read File  ^N Replace    ^U Paste      ^J Justify
^C Location  ^_ Go To Line ^M-U Undo
^M-E Redo
```

- 1) Prima di tutto dobbiamo capire di cosa si tratta e cosa fa questo programma. Possiamo facilmente intuire che si tratta di un Assistente Virtuale impostato per dare le risposte a determinate domande, ovvero:

```
import datetime
python
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."
```

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

Qualora non chiedessimo nessuna di queste scelte opzioni questa Assistente Virtuale ci risponderà stampando il messaggio:

“Non ho capito la tua domanda”

Se invece non avessimo necessità di chiedere queste informazioni potremmo semplicemente digitare “esci” e ci saluterà con un: “Arrivederci!”

Si noti che abbiamo la possibilità di scrivere esci in qualsiasi modo, che sia con maiuscole o minuscole o miste, perché la funzione .lower() in Python restituisce come valore lettere minuscole.

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

Una volta digitato “esci” l’istruzione break interromperà il programma.

2)A questo punto andiamo ad individuare casistiche non standard che il progetto non gestisce come ad esempio comportamenti potenziali non contemplati.

- Nessuna possibilità di errore nel formulare la domanda: pretende che la richiesta sia effettuata esattamente come previsto nel codice.
- Essendo Case Sensitive è sensibile alle maiuscole e minuscole e li considera come caratteri distinti.

- Non è presente una lista di opzioni quindi l'utente dovrà "indovinare" il comando corretto e con la giusta sintassi.
- Continua in loop finchè non si utilizza il comando "esci".

3) Andiamo adesso ad analizzare gli errori di sintassi e logici presenti nel programma:

A: Il primo errore di sintassi che balza agli occhi è presente nella condizione Booleana "while true". Infatti sono assenti i ":" alla fine della riga di codice.

```
while True
```

B: Il secondo errore di sintassi si presenta nella condizione "if", infatti non è corretto scrivere datetime.datetime.today()

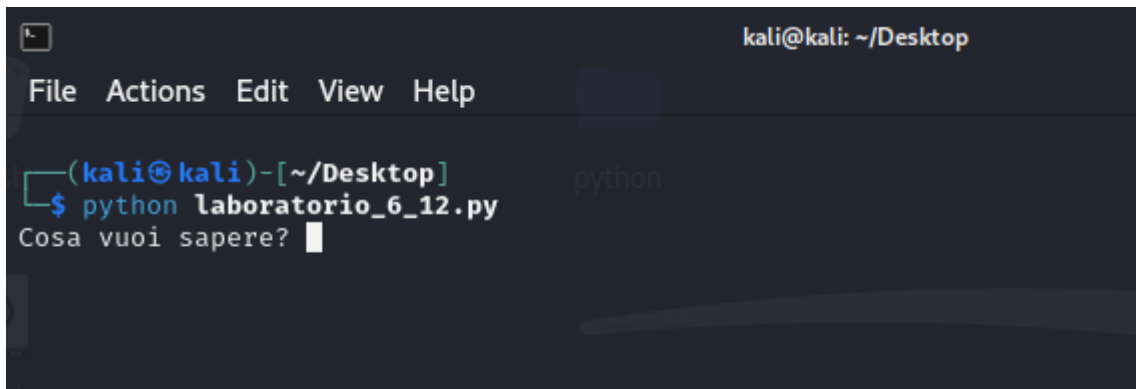
```
if comando == "Qual è la data di oggi?":  
    oggi = datetime.datetime.today()  
    risposta = "La data di oggi è " + oggi.strftime("%d/%m/%Y")
```

C: Il terzo invece è nella condizione "else" in quanto non si scrive datetime.datetime.now().time()

```
elif comando == "Che ore sono?":  
    ora_attuale = datetime.datetime.now().time()  
    risposta = "L'ora attuale è " + ora_attuale.strftime("%H:%M")
```

D: Per quanto riguarda gli errori di logica non sono stampate le opzioni di scelta e risulta quasi impossibile effettuare la giusta

richiesta all' Assistente Virtuale. All' esecuzione del programma infatti verrà chiesto solo "Cosa vuoi sapere?"



```
kali@kali: ~/Desktop
File Actions Edit View Help
(kali@kali)-[~/Desktop]
$ python laboratorio_6_12.py
Cosa vuoi sapere? █
```

E: Il codice è Case Sensitive e non comprende la richiesta effettuata con caratteri scritti genericamente maiuscoli o minuscoli ma solo nell' esatto formato.

F: L' Assistente Virtuale stampa l' orario corrispondente al fuso orario della macchina che lo ha progettato e non in base a quello dell' utente che la richiede.

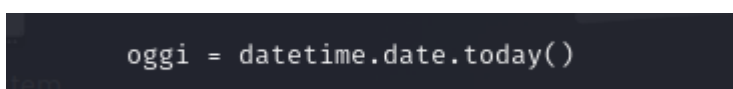
#### 4)Risoluzione errori:

A: Basta inserire ":" alla fine della riga di codice contenente la condizione while True



```
while True:
```

B: Bisogna inserire il "." tra date e today in datetime.date.today()



```
oggi = datetime.date.today()
```

C: Deve essere eliminata la parte “.time()” in `datetime.datetime.now().time()`

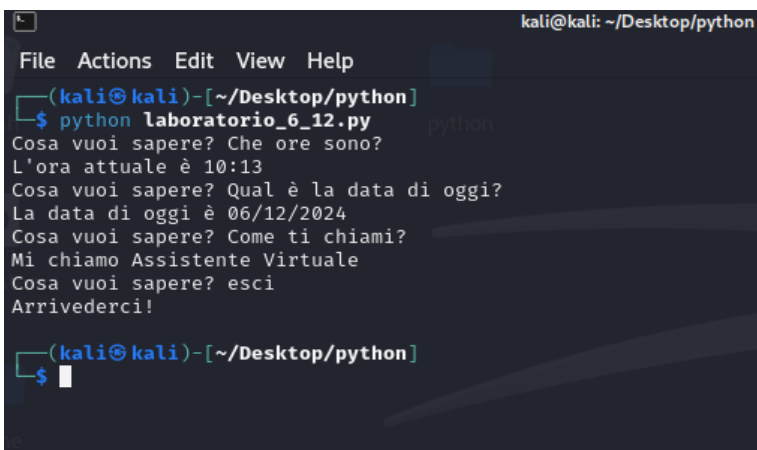
```
ora_attuale = datetime.datetime.now()
```

D: Bisognerebbe stampare una lista delle possibili opzioni per dare la possibilità all’ utente di utilizzarle correttamente.

E: C’è la necessità di inserire una funzione `.lower()` per far sì che il programma non sia più case sensitive e l’utente possa scrivere la propria richiesta indipendentemente se utilizza caratteri maiuscoli o minuscoli.

F: Bisogna aggiungere una opzione che riconosca il fuso orario dell’ utente quando quest’ ultimo formula la richiesta “Che ore sono?” all’ Assistente Virtuale. Questa implica importare anche il pacchetto `pytz` che comprende tutti i fusi orari presenti nel mondo.

Di seguito vediamo l’ Assistente Virtuale in esecuzione:



```
kali@kali: ~/Desktop/python
File Actions Edit View Help
(kali@kali)-[~/Desktop/python]
$ python laboratorio_6_12.py
Cosa vuoi sapere? Che ore sono?
L'ora attuale è 10:13
Cosa vuoi sapere? Qual è la data di oggi?
La data di oggi è 06/12/2024
Cosa vuoi sapere? Come ti chiami?
Mi chiamo Assistente Virtuale
Cosa vuoi sapere? esci
Arrivederci!

(kali@kali)-[~/Desktop/python]
$
```

