

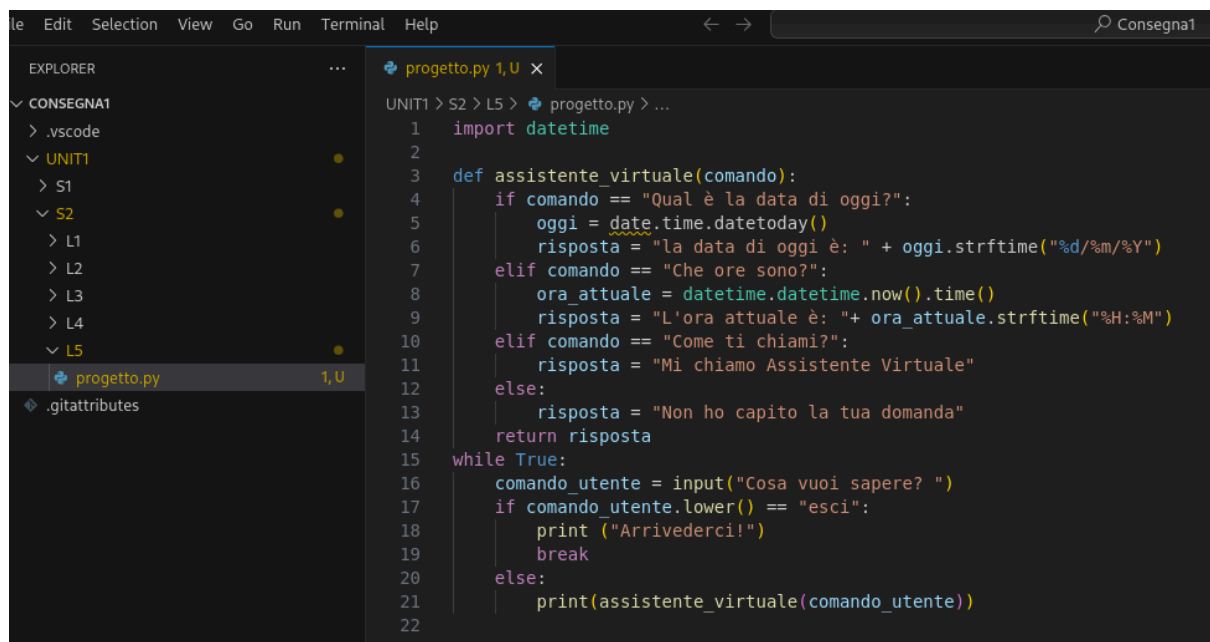
Esercizio per il progetto settimanale Unit 1 - S2 - L5

Svolto da Gioele Parla

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.

Prima di tutto replichiamo il codice su Visual Studio Code all'interno della nostra macchina Kali così potremo analizzarlo più profondamente tramite il terminale



```
le Edit Selection View Go Run Terminal Help
progetto.py 1, U x
UNIT1 > S2 > L5 > progetto.py > ...
1 import datetime
2
3 def assistente_virtuale(comando):
4     if comando == "Qual è la data di oggi?":
5         oggi = date.time.datetoday()
6         risposta = "la data di oggi è: " + oggi.strftime("%d/%m/%Y")
7     elif comando == "Che ore sono?":
8         ora_attuale = datetime.datetime.now().time()
9         risposta = "L'ora attuale è: " + ora_attuale.strftime("%H:%M")
10    elif comando == "Come ti chiami?":
11        risposta = "Mi chiamo Assistente Virtuale"
12    else:
13        risposta = "Non ho capito la tua domanda"
14    return risposta
15 while True:
16    comando_utente = input("Cosa vuoi sapere? ")
17    if comando_utente.lower() == "esci":
18        print("Arrivederci!")
19        break
20    else:
21        print(assistente_virtuale(comando_utente))
22
23
```

Innanzitutto possiamo intuire che Il programma è un assistente virtuale interattivo in cui si chiede all'utente un input e tramite if,elif,else se il comando è:

"Qual è la data di oggi?", risponde con la data odierna.

"Che ore sono?", risponde con l'orario attuale.

"Come ti chiami?", dice il nome dell'assistente.

Altrimenti risponderà sempre con "Non ho capito la tua domanda" e se l'utente scrive "esci" il programma terminerà.

Come prima cosa proviamo ad **avviare** il programma dal terminale per vedere se risponde alla prima domanda, una volta che avremo digitato “Qual è la data di oggi?” sul terminale esso ci restituirà un **errore** dovuto al codice errato che quindi di conseguenza non ci risponderà con la data di oggi.

```
⊗ $ /usr/bin/python /home/kali/Desktop/Code-C/Consegna1/UNIT1/S2/L5/progetto.py
Cosa vuoi sapere? Qual è la data di oggi?
Traceback (most recent call last):
  File "/home/kali/Desktop/Code-C/Consegna1/UNIT1/S2/L5/progetto.py", line 21, in <module>
    print(assistente_virtuale(comando_utente))
    ~~~~~^~~~~~
  File "/home/kali/Desktop/Code-C/Consegna1/UNIT1/S2/L5/progetto.py", line 5, in assistente_virtuale
    oggi = datetime.datetoday()
    ~~~~~^~~~~~
AttributeError: module 'datetime' has no attribute 'datetoday'
```

Per risolvere questo problema dovremo procedere a modificare il nostro codice andando a sostituire **datetime.datetoday** che non esiste con quello corretto: **datetime.date.today** (è importante consultare i comandi corretti di ogni libreria che importiamo)

```
UNIT1 > S2 > L5 > progetto.py > assistente_virtuale
1  import datetime
2
3  def assistente_virtuale(comando):
4      if comando == "qual è la data di oggi?":
5          oggi = datetime.date.today()
```

Una volta corretto il comando per ricevere la giusta risposta, alla prima domanda dell'utente all'interno del terminale: “qual è la data di oggi?” il programma risponderà **correttamente**

```
(kali@kali) - [~/Desktop/Code-C/Consegna1]
$ /usr/bin/python /home/kali/Desktop/Code-C/Consegna1/UNIT1/S2/L5/progetto.py
Cosa vuoi sapere? Qual è la data di oggi?
La data di oggi è: 11/07/2025
```

Per il resto provando a scrivere le altre domande e il programma risponderà a tutto correttamente e con il comando “Esci” chiuderà il programma.

```
(kali㉿kali)-[~/Desktop/Code-C/Consegna1]
• $ /usr/bin/python /home/kali/Desktop/Code-C/Consegna1/UNIT1/S2/L5/progetto.py
Cosa vuoi sapere? Qual è la data di oggi?
La data di oggi è: 11/07/2025
Cosa vuoi sapere? Che ore sono?
L'ora attuale è: 08:45
Cosa vuoi sapere? Come ti chiami?
Mi chiamo Assistente Virtuale
Cosa vuoi sapere? esci
Arrivederci!
```

Per rendere migliore il ciclo dell'assistente virtuale aggiungiamo anche un avviso per quando non rileva un comando valido e così puoi ripeterlo e richiedere il giusto comando, aggiungendo `if not comando_utente.strip.lower()` se l'utente ha premuto solo Invio o ha scritto solo spazi o non ha scritto nulla e preme invio spunterà l'avviso, così non risponderà sempre e solo non ho capito la tua domanda ma di inserire un comando valido.

```
15 while True:
16     comando_utente = input("Cosa vuoi sapere? ")
17     if not comando_utente.strip.lower():
18         print("Per favore inserisci un comando valido.")
19         continue
20     if comando_utente.strip.lower() == "esci":
21         print("Arrivederci!")
22         break
23     else:
24         print(assistente_virtuale(comando_utente))
25
```

Come ultima cosa essendo il programma **case-sensitive** anche solo una maiuscola può confondere il programma quindi possiamo renderlo più flessibile e intuitivo per l'utente eliminando questo “problema” che potrebbe non capire se scriviamo la stessa domanda ma con una maiuscola invece che una minuscola e viceversa, ci basterà aggiungere all'inizio un'impostazione per il comando che leggerà il comando sempre tutto minuscolo per il programma e senza spazi ovvero: **`comando = comando.strip().lower()`**

```
UNIT1 > S2 > L5 > progetto.py > assistente_virtuale
1 import datetime
2
3 def assistente_virtuale(comando):
4     comando = comando.strip().lower()
5     if comando == "Qual è la data di oggi?":
```