ANALISI E RISOLUZIONE DEL CODICE

CODICE DA ANALIZZARE E RISOLVERE

```
import datetime
def assistente_virtuale(comando):
   if comando == "qual'è la data di oggi?":
   oggi = datetime.datetoday()
   risposta = "La data di oggi è " +
oggi.striftime("%d/%m/%Y")
   elif comando == "Che ore sono?":
      ora_attuale = datetime.datetime.now().time()
      risposta = "L'ora attuale è " +
ora_attuale.striftime("%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_utene))
```

- # ANALISI

- In questo programma si vorrebbe programmare un'assistente virtuale a cui chiedere la data e l'ora di oggi, tramite il linguaggio di programmazione python. Nello specifico il programma va ad utilizzare il modulo di un'altro programma (datetime), importando tutte le sue funzioni, classi e variabili. Viene di seguito scritta una funzione (def), che verrà utilizzata per inserire nuovi blocchi di codice suo interno, riutilizzabili a sua volta. All'interno della funzione della "IA" viene gestita tramite ciclo if-else, che organizzera le diverse parti di codice in base al valore di una condizione booleana. Naturlmente nel codice, sono presenti le variabili (personalizzabili) e gli input da dare a la "IA" per rendere il programma interattivo. Di seguito, al di fuori della funzione è presente un'altro ciclo, ovvero il while, quest'ultimo è un blocco di codice che verrà eseguito ripetutamente finchè la condizione è valutata vera (True), il ciclo termina quando la condizione viene valutata falsa (False). Infine si riporta l'istruzione break, che serve per terminare un ciclo while prematuramente: non appena quest'espressione viene letta e processata all'interno del ciclo, Python blocca il loop istantaneamente.

- # =======I> RISOLUZIONE

- RISOLUZIONE

CODICE RISCRITTO

```
def assistente_virtuale():
    from datetime import datetime
    print("-Ciao")
    print("-Sono l'assistente virtuale")
    print("-Chiedimi 'che ora è?' o 'la data di
oggi?','esci' per salutarci..")
    print("-Cosa vuoi sapere?:")
    now = datetime.now()
    scelta = input("")
    if scelta == "che ora è?":
        ora = now.strftime("%H:%M:%S")
        print("In questo momento sono le ore: ", ora)
    elif scelta == "la data di oggi?":
        data = now.strftime("%d/%m/%Y")
        print("Oggi è il: ", data)
    elif scelta == "esci":
        print("Arrivederci")
    else:
        print("Scelta non valida")
assistente_virtuale();
```

- Qui ho personalmete riscritto il codice su python. Riscrivendolo, ho riscrontato degli errori lessicali e logici nel codice originale:
- - Errore nella posizione del modulo, che viene posto prima della funzione, comprendenti di errori lessicali entrambi (from non presente nel modulo);
- - Non presenti prima del ciclo if-else, i vari print di stampa per dare modo di presentare alla IA una sorta di interfaccia utente, che dia modo a quest'ultimo di poter capire anche eventuali input e meccanismi di scelta ad esempio;
- -Nel ciclo if-else, ci sono errori sia nel richiamo al modulo per definire le variabili, sia nei print stampa finali di risposta all'utente, al verificarsi di una condizione
- -Dopo fine funzione viene usato un ciclo while, che a parer mio in questo programma non ritengo che serva alla risoluzione finale di quest'ultimo, comprendente anche quest'ultimo di errori lessicali.

- ==========I> EXTRA CONCLUSIONI

CONCLUSIONI

- In conclusione ho trovato impressionante questo modulo importato nel programma per le moltitudini funzionalità da poter applicare, e la sua complessità. Ho visto anche le innumerevoli directve applicabili per poter cambiare il senso di ogni eventuale specifica richiesta.

- Grazie