

# Progetto Software

## 1. Introduzione

In questo report oggi ci viene chiesto di analizzare un codice, trovare gli errori all'interno del codice e implementare misure che possano migliorare il codice.

Il codice in questione dovrebbe essere in grado di ricreare un assistente virtuale in grado di dirci che ore e che giorno siano;

In calce vi è il codice in questione:

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

## 2. Ricerca degli errori

il primo step riguarda la ricerca degli errori all'interno del codice che possono essere di 3 tipi:

- Sintassi: errori ortografici commessi durante la fase di scrittura del codice

- Logici: errori nella progettazione dell'algoritmo; non vengono identificati dal compilatore o dall'interprete e quindi sono particolarmente insidiosi da affrontare
- Esecuzione: altrettanto insidiosi poiché verranno ugualmente eseguiti.

Una volta compresi i tipi di errori che vengono compiuti all'interno del codice, possiamo provare a cercare gli errori all'interno del codice.

In sostanza, gli errori finora rilevati sono 6

#### 1. Errore in riga 3.

Per poter funzionare, un ciclo ha bisogno necessariamente di essere seguito dai due punti ":".

Ma cos'è un ciclo all'interno di un codice?

Sostanzialmente è una stringa di codice testuale che ci permette di creare delle scelte all'interno del programma; ci sono diversi tipi di ciclo, tra questi ci sono:

- If
- While
- For

#### 2. errore in riga 8:

La funzione break ha lo scopo di terminare il programma una volta rispettati determinati requisiti. Invece in questo caso potrebbe continuare a ciclare il print, poiché l'else si trova sotto di lui

#### 3. errore in riga 9:

Viene inserito nella funzione "print(assistente\_virtuale(comando\_utente));

Probabilmente questo errore di tipo logico, poiché l'insidia si nasconde nella progettazione.

Si può pensare che questo “assistente\_virtuale” dovesse essere definito o come una riga di testo da far uscire all’utente, o come una sorta di variabile, dato che entrambi le locuzioni sono unite dal backslash, tipico, per convenzione, di un tipo di variabile.

#### 4. Errore a riga 13:

Questo errore è ugualmente di tipo sintassi in quanto la funzione che va a richiamare il modulo “datetime.datetime.today()” non esiste, è corretto invece utilizzare datetime.datetime.today poiché ci stiamo riferendo prima alla libreria, poi al modulo dello script, e alla funzione del modulo contenuto nella libreria.

#### 5. Errore in riga 25:

Senza un print, sarà difficile per il nostro programma elaborare una risposta.

#### 6. Def:

Probabilmente questo è un errore logico, in quanto la progettazione non permetteva la corretta esecuzione del programma, trascinando l’utente in un loop infinito.

Per ora, dall’analisi effettuata, per lo più sono errori di sintassi, ma il problema principale resta la progettazione.

### 3. Fix e miglioramento dell’esperienza utente

Innanzitutto, ciò che dovrebbe essere ricordato è che Python è case sensitive, quindi nell’input che si aspetta da noi, devono essere soddisfatti alcuni requisiti.

Per avere una migliore gestione, sarebbe comodo utilizzare delle liste all’interno del ciclo if:

In alternativa, per avere un codice più pulito possiamo fare `from datetime import datetime`.

