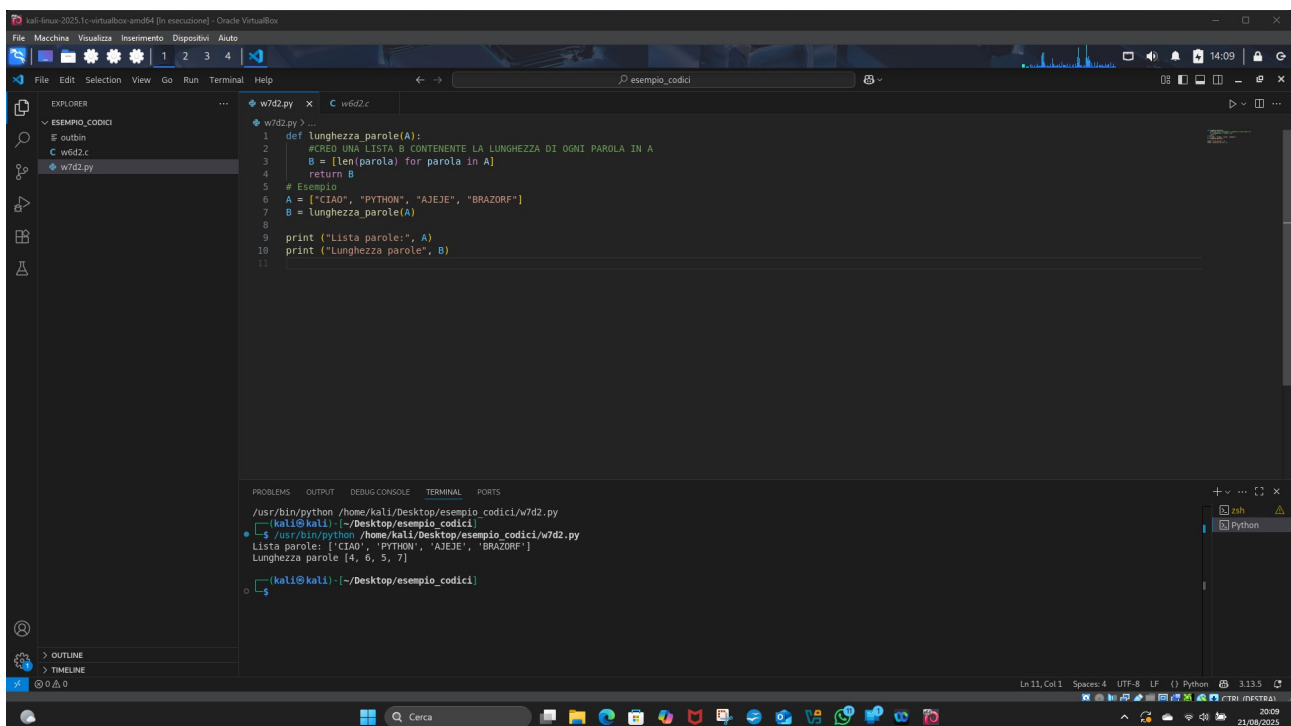


Esercizio W7D2

L'esercizio richiede la creazione tramite Python di una funzione che data una lista A restituisca una lista B di interi che rappresenti la lunghezza di ogni singola parola



The screenshot shows a code editor with a Python script named `w7d2.py`. The script defines a function `lunghezza_parole(A)` that takes a list `A` and returns a list `B` containing the length of each word in `A`. The function is tested with the list `A = ["CIAO", "PYTHON", "AJEJE", "BRAZORF"]`. The output of the script is displayed in the terminal window below the code editor.

```
def lunghezza_parole(A):
    #CREO UNA LISTA B CONTENENTE LA LUNGHEZZA DI OGNI PAROLA IN A
    B = [len(parola) for parola in A]
    return B
# Esempio
A = ["CIAO", "PYTHON", "AJEJE", "BRAZORF"]
B = lunghezza_parole(A)
print ('Lista parole:', A)
print ('Lunghezza parole', B)
```

```
/usr/bin/python /home/kali/Desktop/esempio_codici/w7d2.py
kali@kali:~/Desktop/esempio_codici$
$ /usr/bin/python /home/kali/Desktop/esempio_codici/w7d2.py
Lista parole: ['CIAO', 'PYTHON', 'AJEJE', 'BRAZORF']
Lunghezza parole [4, 6, 5, 7]
kali@kali:~/Desktop/esempio_codici$
```

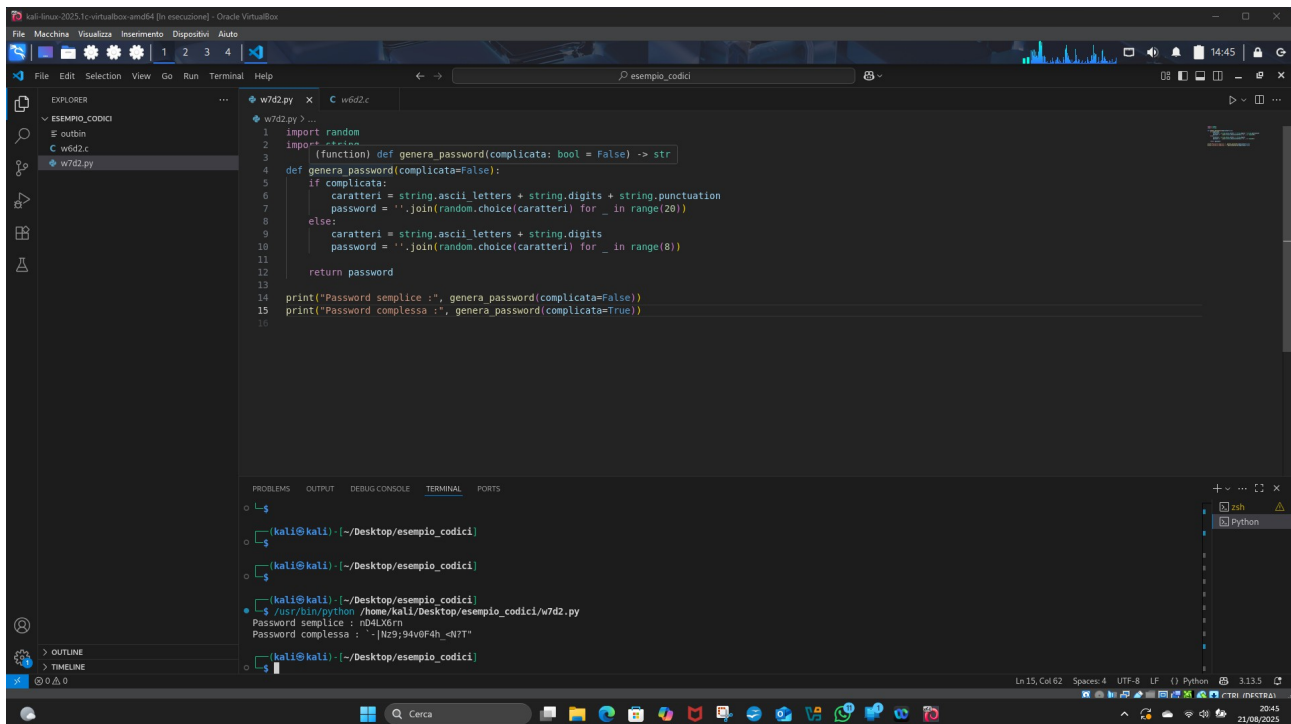
Abbiamo utilizzato `def` per definire la funzione , `lunghezza_parole` e' il nome della funzione ,invece `A` e' il parametro contenente le parole.

Abbiamo creato la lista `B` , `len(parola)` calcola la lunghezza delle parole e `for parola in A` significa che analizza ogni parola contenuta nella lista `A`.

IL RISULTATO FINALE SARA' UNA NUOVA LISTA `B` CONTENENTE TUTTI I NUMERI DELLE PAROLE PRESENTI NELLA LISTA `A`

Esercizio Facoltativo

l'esercizio facoltativo richiede la creazione di un generatore di password sia semplici con 8 caratteri sia complesse con 20 caratteri ASCII casuali.



```
1 import random
2 import string
3 (function) def genera_password(complicata: bool = False) -> str:
4     def genera_password(complicata=False):
5         if complicata:
6             caratteri = string.ascii_letters + string.digits + string.punctuation
7             password = ''.join(random.choice(caratteri) for _ in range(20))
8         else:
9             caratteri = string.ascii_letters + string.digits
10            password = ''.join(random.choice(caratteri) for _ in range(8))
11        return password
12
13 print("Password semplice :", genera_password(complicata=False))
14 print("Password complessa :", genera_password(complicata=True))
15
```

Terminal Output:

```
(kali@kali) ~/Desktop/esempio_codici
$ python w7d2.py
Password semplice : nD4LX6rn
Password complessa : '-|Nz9;94v0F4h_-dN7T'
```

**Per eseguire l'esercizio dobbiamo importare 2 librerie sia random che string
random ci servira' per generare caratteri casuali**

string invece contiene insiemi di caratteri gia' pronti :

string.ascii_letters contiene tutte le lettere maiuscole minuscole

string.digits tutte le cifre da 0 a 9

string.punctuation tutti i simboli speciali

**abbiamo creato una funzione chimata genera_password, ha un parametro
chiamato complicata ed e' impostata su False (riga 4)**

**dalla riga 5 specifichiamo cosa si intende per complicata e nella riga 6 daremo
ordine di inserire numeri lettere e simboli**

**nella riga 7 gli diremo che la creazione deve essere casuale ed essendo complicata
dovra' contenere 20 caratteri**

**nel caso di password semplice il procedimento sara' lo stesso ma con soli 8
caratteri e con lettere e numeri.**