

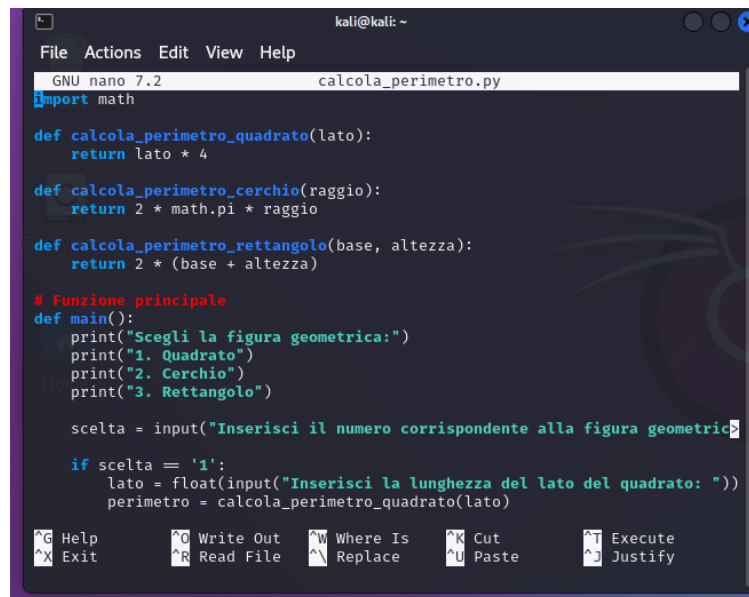
ESERCIZIO W6D4

Traccia: Scrivere un programma in Python che permetta di calcolare il perimetro di diverse figure geometriche in base alla scelta dell'utente. Le figure geometriche considerate sono: Quadrato, Cerchio e Rettangolo.

Metodologia seguita:

1. Apertura dell'Editor di Testo:

- Ho aperto il terminale su Kali Linux.
- Successivamente, ho utilizzato l'editor di testo **nano** per iniziare a scrivere il codice Python.



```
File Actions Edit View Help
GNU nano 7.2 calcola_perimetro.py
import math

def calcola_perimetro_quadrato(lato):
    return lato * 4

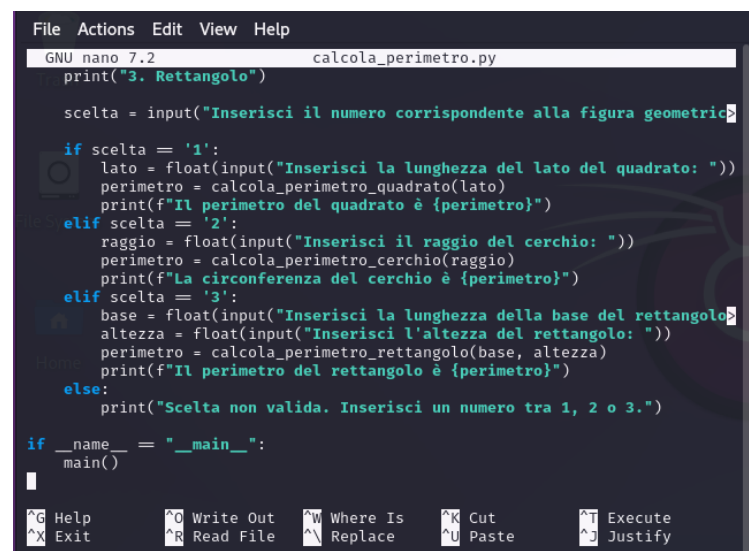
def calcola_perimetro_cerchio(raggio):
    return 2 * math.pi * raggio

def calcola_perimetro Rettangolo(base, altezza):
    return 2 * (base + altezza)

# Funzione principale
def main():
    print("Scegli la figura geometrica:")
    print("1. Quadrato")
    print("2. Cerchio")
    print("3. Rettangolo")

    scelta = input("Inserisci il numero corrispondente alla figura geometrica: ")

    if scelta == '1':
        lato = float(input("Inserisci la lunghezza del lato del quadrato: "))
        perimetro = calcola_perimetro_quadrato(lato)
```



```
File Actions Edit View Help
GNU nano 7.2 calcola_perimetro.py
    print("3. Rettangolo")

    scelta = input("Inserisci il numero corrispondente alla figura geometrica: ")

    if scelta == '1':
        lato = float(input("Inserisci la lunghezza del lato del quadrato: "))
        perimetro = calcola_perimetro_quadrato(lato)
        print(f"Il perimetro del quadrato è {perimetro}")
    elif scelta == '2':
        raggio = float(input("Inserisci il raggio del cerchio: "))
        perimetro = calcola_perimetro_cerchio(raggio)
        print(f"La circonferenza del cerchio è {perimetro}")
    elif scelta == '3':
        base = float(input("Inserisci la lunghezza della base del rettangolo: "))
        altezza = float(input("Inserisci l'altezza del rettangolo: "))
        perimetro = calcola_perimetro Rettangolo(base, altezza)
        print(f"Il perimetro del rettangolo è {perimetro}")
    else:
        print("Scelta non valida. Inserisci un numero tra 1, 2 o 3.")

if __name__ == "__main__":
    main()
```

2. Scrittura del Codice:

- Ho iniziato il codice definendo tre funzioni separate per calcolare il perimetro di Quadrato, Cerchio e Rettangolo.

- Ho implementato la funzione principale (**main**) che guida l'utente nella scelta della figura geometrica e richiede i dati necessari per il calcolo del perimetro.
- Ho utilizzato una struttura di controllo (**if-elif-else**) per gestire la scelta dell'utente e richiamare la funzione appropriata.

3. Visualizzazione e Salvataggio:

- Dopo aver scritto il codice, ho salvato il file con il nome **calcola_perimetro.py** usando il comando **Ctrl + O** in **nano**.
- Successivamente, ho chiuso l'editor con **Ctrl + X**.

4. Esecuzione del Codice:

- Ho eseguito lo script direttamente dal terminale usando il comando **python calcola_perimetro.py**.
- Il programma ha visualizzato correttamente il menu di scelta e ha richiesto i dati all'utente.

```

File Actions Edit View Help
(kali@kali)-[~]
$ python calcola_perimetro.py
Scegli la figura geometrica:
1. Quadrato
2. Cerchio
3. Rettangolo
Inserisci il numero corrispondente alla figura geometrica desiderata: 1
Inserisci la lunghezza del lato del quadrato: 4
Il perimetro del quadrato è 16.0

(kali@kali)-[~]
$ python calcola_perimetro.py
Scegli la figura geometrica:
1. Quadrato
2. Cerchio
3. Rettangolo
Inserisci il numero corrispondente alla figura geometrica desiderata: 2
Inserisci il raggio del cerchio: 3
La circonferenza del cerchio è 18.84955592153876

(kali@kali)-[~]
$ python calcola_perimetro.py
Scegli la figura geometrica:
1. Quadrato
2. Cerchio
3. Rettangolo
Inserisci il numero corrispondente alla figura geometrica desiderata: 3
Inserisci la lunghezza della base del rettangolo: 3
Inserisci l'altezza del rettangolo: 4
Il perimetro del rettangolo è 14.0

(kali@kali)-[~]
$

```

5. Visualizzazione del Risultato:

- Ho inserito i dati richiesti e il programma ha restituito correttamente il perimetro calcolato in base alla figura geometrica scelta.

Considerazioni Finali:

- La soluzione proposta rispetta i requisiti della traccia, consentendo all'utente di calcolare il perimetro di diverse figure geometriche in modo interattivo.

- Il codice è strutturato in modo chiaro e organizzato, facilitando la comprensione e la manutenzione.
- L'utilizzo di funzioni separate per il calcolo del perimetro di ciascuna figura geometrica contribuisce a una buona modularità.

La soluzione proposta ha dimostrato successo nell'affrontare la traccia assegnata, rispettando le specifiche richieste e producendo un programma funzionante.