Primo Programma

November 9, 2023

1 Il Primo Programma

[ ]:
```
print ("Ciao, mondo!")
```

[ ]:
```
nome = input ("Inserisci il tuo nome: ")
print ("Ciao,", nome, "!")
```

[ ]:
```
nome="mattia"
print(nome)
```

[ ]:
```
via=input("Inserisci nome della via")
print("hai inserito",via)
```

[ ]:
```
nome = input("inserisci il tuo nome:")
for i in range(10):
    print("Ciao",nome,"!") #Python ripete solo la parte di codice identata␣
 ↪(ovvero quello che è spostato a destra)
```

2 Operazioni matematiche

3 Somma

numero1 = int(input("Inserisci il primo numero:"))#Int vuol dire numero intero numero2 = int(input("Inserisci il secondo numero:")) somma = numero1 + numero2 print("La somma è:", somma)

4 Sottrazione

[ ]:
```
#Sottrazione
sottrazione = numero1 - numero2
print("La sostrazione è:", int(sottrazione))
```

5 Moltiplicazione

[ ]:
```
numero1 = int(input("Inserisci il primo numero: "))
numero2 = int(input("Inserisci il secondo numero"))
moltiplicazione = numero1 * numero2
print("La moltiplicazione è:", moltiplicazione)
```

6 Divisione

[ ]:
```
divisione= numero1 / numero2
print("La divisione è:",divisione)
```

[ ]:
```

```

[3]:
```
#Loop e ripetizione
for numero in range(1,11):
    print(numero)
```

```
1
2
3
4
5
6
7
8
9
10
```

7 Condizioni e decisioni

8 Calcolatrice di Python

[12]:
```
# Chiede all'utente di inserire l'operazione desiderata (+, -, *, /)
operazione = input("Inserisci l'operazione (+, -, *, /): ")

# Richiede all'utente di inserire il primo numero, trattato come valore in␣
 ↪virgola mobile
numero1 = float(input("Inserisci il primo numero: "))

# Richiede all'utente di inserire il secondo numero, trattato come valore in␣
 ↪virgola mobile
numero2 = float(input("Inserisci il secondo numero: "))

# Utilizza una struttura condizionale per determinare l'operazione e calcolare␣
 ↪il risultato
```

```
if operazione == "+":  # Se l'operazione è l'addizione
    risultato = numero1 + numero2
elif operazione == "-":  # Se l'operazione è la sottrazione
    risultato = numero1 - numero2
elif operazione == "*":  # Se l'operazione è la moltiplicazione
    risultato = numero1 * numero2
elif operazione == "/":  # Se l'operazione è la divisione
    if numero2 != 0:  # Verifica se il denominatore (numero2) è diverso da zero
        risultato = numero1 / numero2
    else:
        risultato = "Impossibile dividere per zero"  # Messaggio di errore se si␣
 ↪cerca di dividere per zero
else:
    risultato = "Operazione non valida"  # Messaggio di errore se l'operazione␣
 ↪non è riconosciuta

# Stampa il risultato dell'operazione
print("Il risultato è:", risultato)
```

```
Inserisci l'operazione (+, -, *, /): +
Inserisci il primo numero: 10
Inserisci il secondo numero: 5
Il risultato è: 15.0
```

9 Confronta fino a n

[10]:
```
n = int(input("Inserisci un numero intero positivo:"))  # Chiede all'utente di␣
 ↪inserire un numero intero positivo e lo memorizza in 'n'
for numero in range(1, n+1):  # Utilizza un ciclo for per iterare da 1 a n␣
 ↪inclusi
    print(numero)  # Stampa il valore corrente del contatore 'numero'
```

```
Inserisci un numero intero positivo:20
1
2
3
4
5
6
7
8
9
10
11
12
13
14
```

15
16
17
18
19
20

10 Somma di numeri positivi

[9]:
```
n = int(input("Inserisci un numero intero positivo:"))  # Chiede all'utente di␣
↪inserire un numero intero positivo e lo memorizza in 'n'
somma = 0  # Inizializza una variabile 'somma' a zero per tenere traccia della␣
↪somma dei numeri
for numero in range(1, n+1):  # Utilizza un ciclo for per iterare da 1 a n␣
↪inclusi
    somma += numero  # Aggiunge il valore corrente di 'numero' alla variabile␣
↪'somma' (equivale a scrivere: somma = somma + numero)
print("La somma dei primi", n, "numeri interi è:", somma)  # Stampa la somma dei␣
↪primi 'n' numeri interi
```

```
Inserisci un numero intero positivo:8
La somma dei primi 8 numeri interi è: 36
```

11 Calcolare il quadrato dei primi numeri

[8]:
```
n = int(input("Inserisci un numero intero positivo: "))  # Chiede all'utente di␣
↪inserire un numero intero positivo e lo memorizza in 'n'
print("Quadrati dei primi", n, "numeri:")  # Stampa un'intestazione per mostrare␣
↪che i seguenti numeri saranno i quadrati dei primi 'n' numeri

for numero in range(1, n + 1):  # Utilizza un ciclo for per iterare da 1 a 'n'␣
↪inclusi
    quadrato = numero ** 2  # Calcola il quadrato del numero corrente e␣
↪memorizzalo in 'quadrato'
    print("Il quadrato di", numero, "è", quadrato)  # Stampa il quadrato del␣
↪numero corrente
```

```
Inserisci un numero intero positivo: 8
Quadrati dei primi 8 numeri:
Il quadrato di 1 è 1
Il quadrato di 2 è 4
Il quadrato di 3 è 9
Il quadrato di 4 è 16
Il quadrato di 5 è 25
Il quadrato di 6 è 36
Il quadrato di 7 è 49
```

```
Il quadrato di 8 è 64
```

12 Verificare la parità

[7]:
```python
numero = int(input("Inserisci un numero:"))  # Chiede all'utente di inserire un numero e lo memorizza in 'numero'

if numero % 2 == 0:  # Controlla se 'numero' è divisibile per 2 senza resto
    print(numero, "è un numero pari.")  # Se il resto della divisione per 2 è 0, allora 'numero' è pari
else:
    print(numero, "è un numero dispari.")  # Altrimenti, 'numero' è dispari
```

```
Inserisci un numero:6
6 è un numero pari.
```

13 Fattoriale

[6]:
```python
n = int(input("Inserisci un numero intero positivo: "))  # Chiede all'utente di inserire un numero intero positivo e lo memorizza in 'n'
fattoriale = 1  # Inizializza una variabile 'fattoriale' a 1

for numero in range(1, n + 1):  # Utilizza un ciclo for per iterare da 1 a n inclusi
    # Calcola il fattoriale moltiplicando il valore corrente di 'numero' per il valore precedente di 'fattoriale'
    fattoriale *= numero
print("Il fattoriale di", n, "è:", fattoriale)  # Stampa il fattoriale del numero 'n'
```

```
Inserisci un numero intero positivo: 8
Il fattoriale di 8 è: 40320
```

14 Calcolare la Media di una lista di Numeri

[5]:
```python
numeri = []  # Crea una lista vuota chiamata 'numeri' per memorizzare i numeri inseriti

n = int(input("Quanti numeri vuoi inserire?"))  # Chiede all'utente quanti numeri vuole inserire e memorizza la risposta in 'n'
for i in range(n):
    numero = float(input("Inserisci un numero: "))  # Chiede all'utente di inserire un numero (come valore in virgola mobile) e lo memorizza in 'numero'
    numeri.append(numero)  # Aggiunge il numero inserito alla lista 'numeri'
```

```
media = sum(numeri) / len(numeri)  # Calcola la media dei numeri nella lista␣
↪dividendo la somma dei numeri per la loro lunghezza

print("La media dei numeri inseriti è:", media, "la lista completa è:", numeri) ␣
↪# Stampa la media dei numeri e la lista completa dei numeri inseriti
```

```
Quanti numeri vuoi inserire?2
Inserisci un numero: 23
Inserisci un numero: 3
La media dei numeri inseriti è: 13.0 la lista completa è: [23.0, 3.0]
```

15 ____

16 Indovina il numero

[4]:
```
import random  # Importa il modulo random per generare numeri casuali

numero_da_indovinare = random.randint(1, 100)  # Genera un numero casuale␣
↪compreso tra 1 e 100 e lo memorizza in 'numero_da_indovinare'
tentativi = 0  # Inizializza una variabile 'tentativi' a 0 per tener traccia del␣
↪numero di tentativi

while True:  # Entra in un ciclo while infinito, poiché non c'è una condizione␣
↪di uscita definita
    tentativo = int(input("Indovina il numero (1-100): "))  # Chiede all'utente␣
↪di indovinare un numero compreso tra 1 e 100 e memorizza il tentativo in␣
↪'tentativo'
    tentativi += 1  # Aumenta il contatore dei tentativi di 1

    if tentativo == numero_da_indovinare:  # Controlla se il tentativo è uguale␣
↪al numero da indovinare
        print("Bravo! Hai indovinato il numero", numero_da_indovinare, "in",␣
↪tentativi, "tentativi.")  # Stampa un messaggio di successo
        break  # Esce dal ciclo while, poiché il numero è stato indovinato

    elif tentativo < numero_da_indovinare:  # Se il tentativo è inferiore al␣
↪numero da indovinare
        print("Il numero è più grande.")  # Stampa un messaggio che suggerisce␣
↪che il numero da indovinare è più grande

    else:  # Se il tentativo non è uguale al numero da indovinare e non è␣
↪minore, significa che è maggiore
        print("Il numero è più piccolo.")  # Stampa un messaggio che suggerisce␣
↪che il numero da indovinare è più piccolo
```

```
Indovina il numero (1-100): 4
Il numero è più grande.
Indovina il numero (1-100): 3
Il numero è più grande.
Indovina il numero (1-100): 78
Il numero è più piccolo.
Indovina il numero (1-100): 50
Il numero è più piccolo.
Indovina il numero (1-100): 30
Il numero è più grande.
Indovina il numero (1-100): 40
Il numero è più piccolo.
Indovina il numero (1-100): 28
Il numero è più grande.
Indovina il numero (1-100): 29
Il numero è più grande.
Indovina il numero (1-100): 35
Il numero è più piccolo.
Indovina il numero (1-100): 33
Il numero è più piccolo.
Indovina il numero (1-100): 32
Bravo! Hai indovinato il numero 32 in 11 tentativi.
```

17 Carta forbice sasso

[15]:
```python
import random  # Importa il modulo random per generare una scelta casuale

mosse = ["carta", "forbice", "sasso"]  # Crea una lista di possibili mosse␣
↪(carta, forbici, sasso)

computer_mossa = random.choice(mosse)  # Il computer fa una scelta casuale tra␣
↪le mosse disponibili

print("Benvenuti al Gioco della Morra Cinese!")
scelta_giocatore = input("Scegli la tua mossa (carta, forbice, sasso):")  #␣
↪Chiede al giocatore di fare una scelta

if scelta_giocatore not in mosse:  # Verifica se la scelta del giocatore è tra␣
↪quelle permesse
    print("Mossa non permessa")
else:
    print("Il computer ha scelto:", computer_mossa)  # Stampa la scelta del␣
↪computer

    if scelta_giocatore == computer_mossa:  # Se la scelta del giocatore è␣
↪uguale a quella del computer
```

```
        print("Pareggio")  # Stampa "Pareggio"

    # Verifica tutte le condizioni in cui il giocatore vince
    elif (scelta_giocatore == "carta" and computer_mossa == "sasso") or \
         (scelta_giocatore == "forbice" and computer_mossa == "carta") or \
         (scelta_giocatore == "sasso" and computer_mossa == "forbice"):
        print("Hai vinto")  # Stampa "Hai vinto"

    # Se nessuna delle condizioni precedenti è vera, il giocatore ha perso
    else:
        print("Hai perso!")  # Stampa "Hai perso"
```

```
Benvenuti al Gioco della Morra Cinese!
Scegli la tua mossa (carta, forbici, sasso):forbice
Il computer ha scelto: carta
Hai vinto
```

18 _____

19 Fattoriale

[16]:
```
n = int(input("Inserisci un numero intero: "))  # Chiede all'utente di inserire␣
↪un numero intero e lo memorizza in 'n'
fattoriale = 1  # Inizializza una variabile 'fattoriale' a 1

if n < 0:  # Controlla se 'n' è un numero negativo
    print("Il numero è negativo.")  # Stampa un messaggio che indica che il␣
↪numero è negativo

elif n == 0:  # Se 'n' è uguale a zero
    print("Il fattoriale di zero è 1.")  # Stampa un messaggio che indica che il␣
↪fattoriale di zero è 1

else:  # Se 'n' è positivo
    for numero in range(1, n + 1):  # Utilizza un ciclo for per iterare da 1 a␣
↪'n' inclusi
        fattoriale *= numero  # Calcola il fattoriale moltiplicando il valore␣
↪corrente del contatore 'numero' per il valore corrente di 'fattoriale'

print(f"Il fattoriale di {n} è {fattoriale}")  # Stampa il risultato, inclusa␣
↪l'interpolazione delle variabili 'n' e 'fattoriale
```

```
Inserisci un numero intero: 20
Il fattoriale di 20 è 2432902008176640000
```

[17]:
```python
n = int(input("Inserisci un numero intero: "))  # Chiede all'utente di inserire un numero intero e lo memorizza in 'n'
fattoriale = 1  # Inizializza una variabile 'fattoriale' a 1

if n < 0:  # Controlla se 'n' è un numero negativo
    print("Numero negativo")  # Stampa un messaggio che indica che il numero è negativo

elif n == 0:  # Se 'n' è uguale a zero
    print("Il fattoriale di zero è 1 per definizione")  # Stampa un messaggio che spiega che il fattoriale di zero è 1 per definizione

else:  # Se 'n' è un numero positivo
    for numero in range(1, n + 1):  # Utilizza un ciclo for per iterare da 1 a 'n' inclusi
        fattoriale *= numero  # Calcola il fattoriale moltiplicando il valore corrente del contatore 'numero' per il valore corrente di 'fattoriale'

print(f"Il fattoriale di {n} è {fattoriale}")  # Stampa il risultato, includendo l'interpolazione delle variabili 'n' e 'fattoriale
```

```
Inserisci un numero intero: 20
Il fattoriale di 20 è 2432902008176640000
```

[ ]:
```python
# Chiedere all'utente di inserire un numero intero positivo N
N = int(input("Inserisci un numero intero positivo N:"))

# Inizializzare la somma a zero
somma = 0

# Calcolare la somma dei primi N numeri pari
for numero in range(2, 2 * N + 1, 2):
    somma += numero

print(f"La somma dei primi {N} numeri pari è {somma}")
```

[19]:
```python
#Chiedere all'utente di inserire un numero intero positivo N
N = int(input("Inserisci un numero intero positivo N: "))
lista=[]

#Calcolare la somma dei primi N numeri pari
for numero in range(2, 2 * N +1, 2):
    lista.append(numero)

print(lista)
```

```
Inserisci un numero intero positivo N: 5
[2, 4, 6, 8, 10]
```

20 Conteggio delle vocali

[39]:
```
#Chiedi all'utente di inserire una frase o una parola
frase = input ("Inserisci una frase o una parola: ").lower() #Converti tutto in
#Lower =Serve per ridurre le stringe di testo minuscole
#Inizializza il contatore delle vocali
conteggio_vocali = 0

#Definisci le vocali da cercare
vocali ="aeiou"

#Scansiona ogni carattere nella frase
for carattere in frase:
    #Verifica se il carattere è una vocale
    if carattere in vocali:
        conteggio_vocali += 1

#Stampa il conteggio delle vocali
print (f"Nella frase inserita ci sono {conteggio_vocali} vocali.")
```

```
Inserisci una frase o una parola: miao
Nella frase inserita ci sono 3 vocali.
```

21 Indovina il numero del dado

[ ]:
```
import random
# Genera un numera da 1 a 6 (simulando il lancio di un dado)
numero_dado =random.randint(1, 6)

#Chiedi all'utente di indovinare il numero
indovina =int(input("Indovina il numero del dado (da 1 a 6): "))

#Verifica se l'utente ha indovinato correttamente
if indovina <1 or indovina >6:
    print("numero non ammesso")
elif indovina ==numero_dado:
    print(f"Complimenti! Il numero del dado era {numero_dado}. Hai indovinato!")
else:
    print(f"Mi dispiace, il numero del dado era {numero_dado}. Meglia fortuna␣
 ↪alla prossima!")
```

22 Data e ora

[ ]:
```
import datetime

today = datetime.datetime.today()
```

```
print (f"oggi è il giorno: {today: %d  %m  %y} ore: {today: %H %M %S}")
```

[ ]:
```
#Inizializza la popolazione e gli anni
popolazione=int(input("Inserire la popolazione iniziale: "))
anni=int(input("Inserire il numero degli anni da simulare: "))
#Tasso di natalità e di mortalità (percentuale annua)
tasso_natalità=float(input("Inserisci il tasso di natalità: "))
tasso_mortalità=float(input("Inserisci il tasso di mortalità: "))

#Simulazione della crescita della popolazione
for anno in range(anni):
    nascite = int(popolazione*tasso_natalità)
    morti = int(popolazione*tasso_mortalità)
    popolazione += (nascite-morti)
    print(f"Anno {anno + 1}: popolazione={int(popolazione)}") #se il print non è␣
 ↪dentro la dentazione da direttamnete il risultato finale
print("Simulazione completata")
```

23 Convertitore di unità di misura

[20]:
```
print("Benvenuto nel convertitore di Unità di Misura!")

# Chiede all'utente cosa desidera convertire (metri, piedi, chilogrammi, libbre)␣
 ↪e converte la risposta in minuscolo
scelta = input("Cosa desideri convertire? (metri/piedi/chilogrammi/libbre)").
 ↪lower()

if scelta == "metri":  # Se l'utente vuole convertire da metri a piedi
    valore = float(input("Inserisci il valore in metri"))  # Chiede all'utente␣
 ↪di inserire il valore in metri e lo converte in un valore in virgola mobile
    risultato = valore * 3.28084  # Esegue la conversione da metri a piedi
    print(f"{valore} metri corrispondono a {risultato} piedi.")  # Stampa il␣
 ↪risultato della conversione

elif scelta == "piedi":  # Se l'utente vuole convertire da piedi a metri
    valore = float(input("Inserisci il valore in piedi: "))  # Chiede all'utente␣
 ↪di inserire il valore in piedi e lo converte in un valore in virgola mobile
    risultato = valore / 3.28084  # Esegue la conversione da piedi a metri
    print(f"{valore} piedi corrispondono a {risultato} metri.")  # Stampa il␣
 ↪risultato della conversione

elif scelta == "chilogrammi":  # Se l'utente vuole convertire da chilogrammi a␣
 ↪libbre
    valore = float(input("Inserisci il valore in chilogrammi"))  # Chiede␣
 ↪all'utente di inserire il valore in chilogrammi e lo converte in un valore in␣
 ↪virgola mobile
```

```
    risultato = valore * 2.20462  # Esegue la conversione da chilogrammi a libbre
    print(f"{valore} chilogrammi corrispondono a {risultato} libbre.")  # Stampa␣
↪il risultato della conversione

elif scelta == "libbre":  # Se l'utente vuole convertire da libbre a chilogrammi
    valore = float(input("Inserisci il valore in libbre: "))  # Chiede␣
↪all'utente di inserire il valore in libbre e lo converte in un valore in␣
↪virgola mobile
    risultato = valore / 2.20462  # Esegue la conversione da libbre a chilogrammi
    print(f"{valore} libbre corrispondono a {risultato} chilogrammi.")  # Stampa␣
↪il risultato della conversione

else:
    print("Scelta non valida. Scegli tra 'metri', 'piedi', 'chilogrammi' o␣
↪'libbre'.")  # Messaggio di errore se l'utente ha inserito una scelta non␣
↪valida
```

```
Benvenuto nel convertitore di Unità di Misura!
Cosa desideri convertire? (metri/piedi/chilogrammi/libbre)libbre
Inserisci il valore in libbre: 10
10.0 libbre corrispondono a 4.535929094356398 chilogrammi.
```

24 Fibonacci

[21]:
```
# Chiede all'utente di inserire un numero n per calcolare l'n-esimo numero di␣
↪Fibonacci
n = int(input("Inserisci un numero n per calcolare l'n-esimo numero di␣
↪Fibonacci"))

# Inizializza le variabili per i primi due numeri di Fibonacci
a = 0
b = 1
c = 1

# Calcola l'n-esimo numero di Fibonacci
if n <= 0:  # Se l'utente ha inserito un numero non positivo
    print("Il numero deve essere maggiore di zero.")  # Stampa un messaggio di␣
↪errore

elif n == 1:  # Se l'utente vuole il primo numero di Fibonacci (caso base)
    risultato = a  # Il risultato è 0

else:
    for iterazione in range(n - 2):  # Utilizza un ciclo for per calcolare␣
↪l'n-esimo numero di Fibonacci
        a = b
```

```
        b = c
        c = a + b  # Aggiorna i valori delle variabili in base alla sequenza di␣
↪Fibonacci

    risultato = c  # Il risultato è il n-esimo numero di Fibonacci

# Stampa l'n-esimo numero di Fibonacci
print("L'n-esimo numero di Fibonacci è:", risultato)
```

```
Inserisci un numero n per calcolare l'n-esimo numero di Fibonacci40
L'n-esimo numero di Fibonacci è: 102334155
```

25 Funzioni custom

[22]:
```
def Fibonacci(n):
    # Definizione di una funzione per calcolare una serie di Fibonacci fino␣
↪all'n-esimo termine

    fib_series = [0, 1]  # Inizializza una lista con i primi due numeri di␣
↪Fibonacci

    while len(fib_series) < n:
        # Continua ad aggiungere numeri alla lista finché la sua lunghezza è␣
↪inferiore a 'n'

        fib_series.append(fib_series[-1] + fib_series[-2])
        # Aggiunge il numero successivo nella sequenza di Fibonacci sommando gli␣
↪ultimi due numeri nella lista

    return fib_series
    # Restituisce la lista completa dei numeri di Fibonacci fino all'n-esimo␣
↪termine
```

[23]:
```
Fibonacci (10)
```

[23]:
```
[0, 1, 1, 2, 3, 5, 8, 13, 21, 34]
```

[24]:
```
n = int(input("Inserisci il numero di termini della serie di Fibonacci"))  #␣
↪Chiede all'utente di inserire il numero di termini desiderati nella serie di␣
↪Fibonacci e lo memorizza in 'n'

if n <= 0:  # Verifica se 'n' è non positivo (zero o negativo)
    print("Inserisci un numero positivo.")  # Stampa un messaggio di errore se␣
↪'n' non è positivo

else:
```

```
    result = Fibonacci(n)  # Chiama una funzione chiamata 'Fibonacci' per␣
↪calcolare la serie di Fibonacci fino all'n-esimo termine
    print(result)  # Stampa la serie di Fibonacci risultante
```

```
Inserisci il numero di termini della serie di Fibonacci4
[0, 1, 1, 2]
```

26 Calcolatrice di aree

[25]:
```
import math  # Importa la libreria math di Python, che contiene funzioni␣
↪matematiche avanzate

def calcola_area_cerchio(raggio):
    # Definizione di una funzione per calcolare l'area di un cerchio
    return math.pi * (raggio ** 2)  # Restituisce l'area del cerchio utilizzando␣
↪la costante matematica 'pi' (π) di math

def calcola_area_rettangolo(base, altezza):
    # Definizione di una funzione per calcolare l'area di un rettangolo
    return base * altezza  # Restituisce l'area del rettangolo moltiplicando␣
↪base e altezza

def calcola_area_triangolo(base, altezza):
    # Definizione di una funzione per calcolare l'area di un triangolo
    return (base * altezza) / 2  # Restituisce l'area del triangolo␣
↪moltiplicando base e altezza e dividendo per 2
```

[ ]:
```
print("Benvenuto nella Calcolatrice di Aree!")  # Stampa un messaggio di␣
↪benvenuto

scelta = input("Vuoi calcolare l'area di un cerchio (c), rettangolo (r) o␣
↪triangolo (t)").lower()
# Chiede all'utente di scegliere quale tipo di area vuole calcolare (cerchio,␣
↪rettangolo o triangolo) e memorizza la scelta in 'scelta', convertendo il␣
↪testo in minuscolo

if scelta == 'c':  # Se l'utente ha scelto di calcolare l'area di un cerchio
    raggio = float(input("Inserisci il raggio del cerchio: "))  # Chiede␣
↪all'utente di inserire il raggio del cerchio e lo converte in un valore in␣
↪virgola mobile
    area = calcola_area_cerchio(raggio)  # Chiama la funzione␣
↪'calcola_area_cerchio' per calcolare l'area del cerchio
    print(f"L'area del cerchio è {area:.2f}")  # Stampa l'area del cerchio con␣
↪due decimali

elif scelta == 'r':  # Se l'utente ha scelto di calcolare l'area di un rettangolo
```

```
    base = float(input("Inserisci la base del rettangolo: "))  # Chiede␣
↪all'utente di inserire la base del rettangolo e lo converte in un valore in␣
↪virgola mobile
    altezza = float(input("Inserisci l'altezza del rettangolo: "))  # Chiede␣
↪all'utente di inserire l'altezza del rettangolo e lo converte in un valore in␣
↪virgola mobile
    area = calcola_area_rettangolo(base, altezza)  # Chiama la funzione␣
↪'calcola_area_rettangolo' per calcolare l'area del rettangolo
    print(f"L'area del rettangolo è {area:.2f}")  # Stampa l'area del rettangolo␣
↪con due decimali

elif scelta == 't':  # Se l'utente ha scelto di calcolare l'area di un triangolo
    base = float(input("Inserisci la base del triangolo: "))  # Chiede␣
↪all'utente di inserire la base del triangolo e lo converte in un valore in␣
↪virgola mobile
    altezza = float(input("Inserisci l'altezza del triangolo: "))  # Chiede␣
↪all'utente di inserire l'altezza del triangolo e lo converte in un valore in␣
↪virgola mobile
    area = calcola_area_rettangolo(base, altezza)  # Chiama la funzione␣
↪'calcola_area_rettangolo' per calcolare l'area del triangolo (potrebbe essere␣
↪un errore di battitura, dovrebbe essere 'calcola_area_triangolo')
    print(f"L'area del triangolo è {area:.2f}")  # Stampa l'area del triangolo␣
↪con due decimali

else:
    print("Scelta non valida. Si prega di inserire 'c', 'r' o 't'.")  #␣
↪Messaggio di errore se l'utente ha inserito una scelta non valida
```

```
Benvenuto nella Calcolatrice di Aree!
```

27 Calcolatore di interessi

[27]:
```
# Questa è una funzione che calcola l'importo finale di un investimento
# basato sull'importo iniziale, il tasso di interesse e il numero di periodi di␣
↪investimento.
def calcola_interessi(importo_iniziale, tasso_interesse, periodi_investimento):
    # Calcola l'importo finale utilizzando la formula dell'interesse composto.
    importo_finale = importo_iniziale * (1 + tasso_interesse / 100) **␣
↪periodi_investimento

    # Restituisce l'importo finale calcolato.
    return importo_finale
```

[28]:
```
#Stampa un messaggio di benvenuto
print("Benvenuto nel Calcolatore Interessi")
```

```
#Definisci una funzione per calcolare l'importo finale con gli interessi
def calcola_interessi(importo, tasso, periodo):
    # Calcola l'importo finale con gli interessi usando la formula degli␣
 ↪interessi composti
    importo_finale = importo * (1 + tasso / 100) ** periodo
    return importo_finale

#Chiedi all'utente di inserire l'importo iniziale, il tasso di interesse e il␣
 ↪periodo di investimento
importo = float(input("Inserisci l'importo iniziale: "))
tasso = float(input("Inserisci il tasso di interesse annuale (%): "))
periodo = int(input("Inserisci il periodo di investimento (anni): "))

#Chiama la funzione calcola_interessi per calcolare l'importo finale
importo_finale = calcola_interessi(importo, tasso, periodo)

#Stampa l'importo finale formattato con due decimali
print(f"L'importo finale dopo {periodo} anni è di {importo_finale:.2f} euro.")
```

```
Benvenuto nel Calcolatore Interessi
Inserisci l'importo iniziale: 30
Inserisci il tasso di interesse annuale (%): 2
Inserisci il periodo di investimento (anni): 5
L'importo finale dopo 5 anni è di 33.12 euro.
```

28 Forza gravitazzionale

[29]:
```
def calcola_forza_gravitazionale(massa1, massa2, distanza):
    # Definizione di una funzione per calcolare la forza gravitazionale tra due␣
 ↪masse e una distanza

    G = 6.67430e-11  # Dichiarazione della costante gravitazionale in N(m/kg)^2

    forza_gravitazionale = (G * massa1 * massa2) / (distanza**2)
    # Calcolo della forza gravitazionale utilizzando la formula F = (G * m1 *␣
 ↪m2) / r^2

    return forza_gravitazionale
    # Restituisce il valore della forza gravitazionale calcolata
```

[30]:
```
# Esempio di utilizzo

massa_terra = 5.972e24  # Massa della Terra in chilogrammi
massa_luna = 7.342e22  # Massa della Luna in chilogrammi
distanza_terra_luna = 384400000  # Distanza tra la Terra e la Luna in metri
```

```
forza = calcola_forza_gravitazionale(massa_terra, massa_luna,␣
 ↪distanza_terra_luna)
# Chiama la funzione 'calcola_forza_gravitazionale' con le masse e la distanza␣
 ↪specificate per calcolare la forza gravitazionale

print(f"Forza gravitazionale tra la Terra e la Luna: {forza} Newton")
# Stampa la forza gravitazionale calcolata tra la Terra e la Luna con il␣
 ↪risultato fornito dalla funzione
```

```
Forza gravitazionale tra la Terra e la Luna: 1.9804922390990566e+20 Newton
```

29 Anagrammi

[ ]:
```
from itertools import permutations

k = 0

def trova_anagrammi(parola):
    anagrammi = ["".join(p) for p in permutations(parola)]
    return anagrammi
```

[ ]:
```
print("Benvenuto nel Risolutore di Anagrammi!")

parola_input = input("Inserisci una parola: ").strip().lower()

if len(parola_input) < 2:
    print("Inserisci una parola con almeno 2 caratteri.")
else:
    anagrammi = trova_anagrammi(parola_input)

    for elemento in anagrammi:
        if elemento != parola_input:
            k += 1
            print(elemento)

    print(f"Gli anagrammi di '{parola_input}' sono: {k}")
```

[ ]:
```
# Chiedi sll'utente di inserire una frase
frase = input("Inserisci una frase: ")

#Converti la frase in minusciki per evitare problemi di maiuscole/minuscole
frase = frase.lower()

#Inizzializza una lista di lettere dell'alfabeto
alfabeto = 'abcdefghijklmnopqrstuvwxyz'

#Inizializza un dizzionario per tenere traccia del conteggio delle lettere
```

```
conteggio_lettere = {}

#Itera attraverso ciascuna lettera dell'alfabeto
for lettera in alfabeto:
    #Conta quante volte appare la lettera nella frase
    conteggio = frase.count(lettera)

    #Aggiungi la lettera e il conteggio al dizionario se la lettera appare
    if conteggio > 0:
        conteggio_lettere[lettera] = conteggio
#Stampa il conteggio delle lettere in un formato leggibiile
for lettera, conteggio in conteggio_lettere.items():
    print(f"{lettera}: {conteggio} ")
```

[ ]:
```
conteggio_lettere.items()
```

[ ]:
```
Prodotti={}
Prodotti["pan bauletto"]=2
Prodotti["coca cola"]=3
```

[ ]:
```
Prodotti
```

30 Tassi di cambio

[ ]:
```
# Definizione dei tassi di cambio
tassi_di_cambio = {
    "dollari": 1.0,
    "euro": 0.85,
    "yen": 110.41,
    # Aggiungi altre valute e tassi di cambio se necessario
}

# Chiedi all'utente di inserire l'importo, la valuta di partenza e la valuta di␣
 ↪destinazione
try:
    importo = float(input("Inserisci l'importo da convertire: "))
    valuta_di_partenza = input("Inserisci la valuta di partenza: ").lower()
    valuta_destinazione = input("Inserisci la valuta di destinazione: ").lower()

    # Verifica se le valute sono nel dizionario dei tassi di cambio
    if valuta_di_partenza in tassi_di_cambio and valuta_destinazione in␣
 ↪tassi_di_cambio:
        # Calcola il tasso di cambio e l'importo convertito
        tasso_di_cambio = tassi_di_cambio[valuta_destinazione] /␣
 ↪tassi_di_cambio[valuta_di_partenza]
        importo_convertito = importo * tasso_di_cambio
```

```
        # Stampa il risultato
        print(f"{importo} {valuta_di_partenza} sono equivalenti a␣
↪{importo_convertito:.2f} {valuta_destinazione}")
    else:
        print("Valute non supportate. Assicurati di inserire valute valide.")
except ValueError:
    print("Inserisci un importo valido.")
```

31 Orologio mondiale

[ ]:
```
from datetime import datetime
import pytz

print("Benvenuto nell'Orologio mondiale!")

# Definisci le città e i relativi fusi orari
citta_fusi_orari = {
    "New York": "America/New_York",
    "Londra": "Europe/London",
    "Tokyo": "Asia/Tokyo",
    "Sydney": "Australia/Sydney",
    "Rio de Janeiro": "America/Sao_Paulo",
}

while True:
    print("\nCittà disponibili: ")
    for citta in citta_fusi_orari.keys():
        print(citta)

    scelta_citta = input("Inserisci il nome della città per visualizzare l'ora␣
↪(o 'esci' per uscire): ").strip()
    if scelta_citta.lower() == 'esci':
        break

    if scelta_citta in citta_fusi_orari:
        fuso_orario = pytz.timezone(citta_fusi_orari[scelta_citta])
        ora_corrente = datetime.now(fuso_orario)
        print(f"L'ora corrente a {scelta_citta} è: {ora_corrente.strftime('%H:%M:
↪%S')}")
    else:
        print("Città non valida. Riprova.")
```

32 Funzione main

[ ]:
```
#Funzione principale può avere qualsiasi nome
def paolo():
    print("Mi chaimo Paolo")
```

[ ]:
```
if __name__ == "__main__": #è una condizione logica che "si verifica sempre"

    paolo()
```

[ ]:
```
#Funzione principale può avere qualsiasi nome
def main():
    print(f"la funzione principale del codice è stata seguita, in questa␣
 ↪funzione possono essere presenti precedentemente create")

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

[ ]:
```
#main
#Funzione per il calcolo del BMI

def calcola_bmi(peso, altezza):
    return peso / (altezza ** 2)

#Funzione per la valutazione del BMI

def valuta_bmi(bmi):
    if bmi < 18.5:
        return "Sottopeso"
    elif 18.5 <= bmi < 24.9:
        return "Normapeso"
    elif 25 <= bmi < 29.9:
        return "Sovrappeso"
    else:
        return "Obeso"
```

33 Calcolo del Bmi

[ ]:
```
#Funzione principale
def main():
    n = int(input("Inserisci numero di persone da valutare: "))
    for persone in range(n):
        print("Benvenuto nella Calcolatrice BMI!")
        peso = float(input("Inserisci il tuo peso in chilogrammi: "))
        altezza = float(input("Inserisci la tua altezza in metri: "))

        bmi = calcola_bmi(peso,altezza)
```

```
        valutazione = valuta_bmi(bmi)

        print(f"Il tuo Bmi è {bmi}")
        print(f"Valutazione: {valutazione}")

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

[1]:
```
# Funzione per la conversione da metri a piedi
def metri_a_piedi(metri):
    return metri * 3.28084

# Funzione per la conversione da piedi a metri
def piedi_a_metri(piedi):
    return piedi / 3.28084

# Funzione per la conversione da chilogrammi a libbre
def chilogrammi_a_libbre(chilogrammi):
    return chilogrammi * 2.20462

# Funzione per la conversione da libbre a chilogrammi
def libbre_a_chilogrammi(libbre):
    return libbre / 2.20462

def selezione(scelta):
    if scelta == "metri":
        valore = float(input("Inserisci il valore in metri: "))
        risultato = metri_a_piedi(valore)
        print(f"{valore:.3f} metri corrispondono a {risultato:.3f} piedi.")
    elif scelta == "piedi":
        valore = float(input("Inserisci il valore in piedi: "))
        risultato = piedi_a_metri(valore)
        print(f"{valore:.3f} piedi corrispondono a {risultato:.3f} metri.")
    elif scelta == "chilogrammi":
        valore = float(input("Inserisci il valore in chilogrammi: "))
        risultato = chilogrammi_a_libbre(valore)
        print(f"{valore:.3f} chilogrammi corrispondono a {risultato:.3f} libbre.
 ↪")
    elif scelta == "libbre":
        valore = float(input("Inserisci il valore in libbre: "))
        risultato = libbre_a_chilogrammi(valore)
        print(f"{valore:.3f} libbre corrispondono a {risultato:.3f} chilogrammi.
 ↪")
    else:
        print("Scelta non valida. Scegli tra 'metri', 'piedi', 'chilogrammi' o␣
 ↪'libbre'.")
```

```
unita_misura = input("Vuoi convertire da metri a piedi, da piedi a metri, da␣
↪chilogrammi a libbre o da libbre a chilogrammi? ").strip().lower()
selezione(unita_misura)
```

```
Vuoi convertire da metri a piedi, da piedi a metri, da chilogrammi a libbre o da
libbre a chilogrammi? piedi
Inserisci il valore in piedi: 100
100.000 piedi corrispondono a 30.480 metri.
```

[ ]:
```
# Dizionario con le calorie per 100 grammi di cibo
cibo_calorie = {
    "banana": 89,
    "mela": 52,
    "arancia": 43,
    # Altri cibi...
}

# Funzione per calcolare le calorie consumate
def calorie_consumate(cibo, quantita):
    if cibo not in cibo_calorie:
        print("Cibo non presente")
        return 0  # Ritorna 0 calorie se il cibo non è nel dizionario
    calorie_per_100g = cibo_calorie[cibo]
    calorie_totali = (calorie_per_100g / 100) * quantita
    return calorie_totali

# Funzione principale
def main():
    cibo_consumato = []

    while True:
        print("Menu")
        print("\n1. Aggiungi cibo consumato")
        print("2. Calcola calorie totali")
        print("3. Esci")

        scelta = input("Scegli un'opzione: ")

        if scelta == "1":
            print("\nCibi disponibili:")
            for cibo in cibo_calorie:
                print(cibo)
            cibo = input("Inserisci il cibo consumato: ").lower()
            quantita = float(input("Inserisci la quantita (in grammi): "))
            cibo_consumato.append((cibo, quantita))

        elif scelta == "2":
```

```
            calorie_totali = sum(calorie_consumate(c, q) for c, q in␣
 ↪cibo_consumato)
            print(f"\nCalorie totali consumate: {calorie_totali} calorie")

        elif scelta == "3":
            break

        else:
            print("\nScelta non valida. Riprova.")

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

[2]:
```
acquisti={}
acquisti ["pan bauletto"]= 10
acquisti ["nutella"]= 10
```

[3]:
```
acquistidue= {
    "pan bauletto":10,
    "nutella":10,
}
```

34 Generatore di personaggi

[4]:
```
import random

# Liste di specie, classi, armi e abilità
speci = ["Elfo", "Umano", "Nano", "Orco", "Gnomo"]
classi = ["Guerriero", "Mago", "Ranger", "Ladro", "Chierico"]
armi = ["Spada", "Arco", "Bacchetta magica", "Ascia", "Daga"]
abilita = ["Furtività", "Magia dell'acqua", "Camuffamento", "Estrazione␣
 ↪mineraria", "Incantesimi di guarigione"]

# Genera un personaggio casuale
specie = random.choice(speci)
classe = random.choice(classi)
arma = random.choice(armi)
abilita_scelte = random.sample(abilita, random.randint(1, 3))

# Stampa il personaggio generato
print("Personaggio Fantasy Generato:")
print(f"Specie: {specie}")
print(f"Classe: {classe}")
print(f"Arma: {arma}")
print(f"Abilità: {', '.join(abilita_scelte)}")
```

```
Personaggio Fantasy Generato:
```

```
Specie: Umano
Classe: Chierico
Arma: Bacchetta magica
Abilità: Magia dell'acqua, Estrazione mineraria
```

[6]:
```python
import random

# Estendi queste liste con attributi, tratti di personalità, sfondi e motivazioni
physicial_traits = ["alto", "basso", "magro", "corpulento", "capelli lunghi",␣
 ↪"capelli corti"]
personality_traits = ["gentile", "introverso", "estroverso", "avventuroso",␣
 ↪"analitico"]
backgrounds = ["contadino", "nobile", "commerciante", "ladro", "mago"]
motivations = ["vendicare una perdita", "trovare l'amore", "arricchirsi",␣
 ↪"scoprire la verità"]

def genera_personaggio():
    nome = input("Inserisci il nome del personaggio: ")
    aspetto_fisico = random.choice(physicial_traits)
    aspetto_personale = random.choice(personality_traits)
    sfondo = random.choice(backgrounds)
    motivazione = random.choice(motivations)

    descrizione = f"{nome} è un personaggio {aspetto_fisico},␣
 ↪{aspetto_personale}, di sfondo {sfondo} e con la motivazione di {motivazione}."

    print(descrizione)

genera_personaggio()
```

```
Inserisci il nome del personaggio: Miao
Miao è un personaggio magro, gentile, di sfondo nobile e con la motivazione di
scoprire la verità.
```