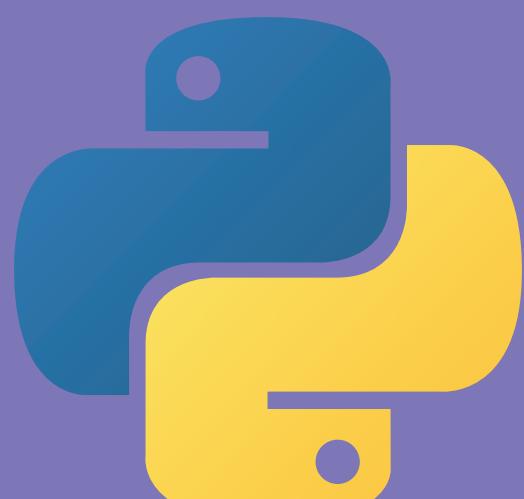


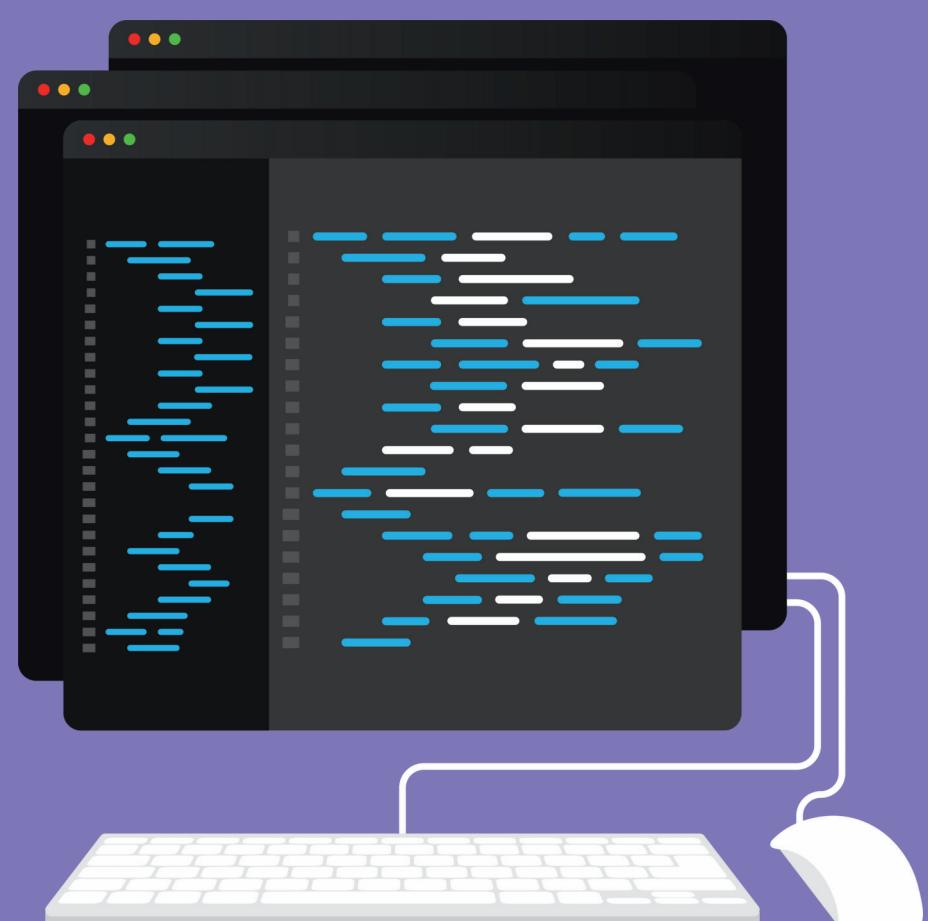
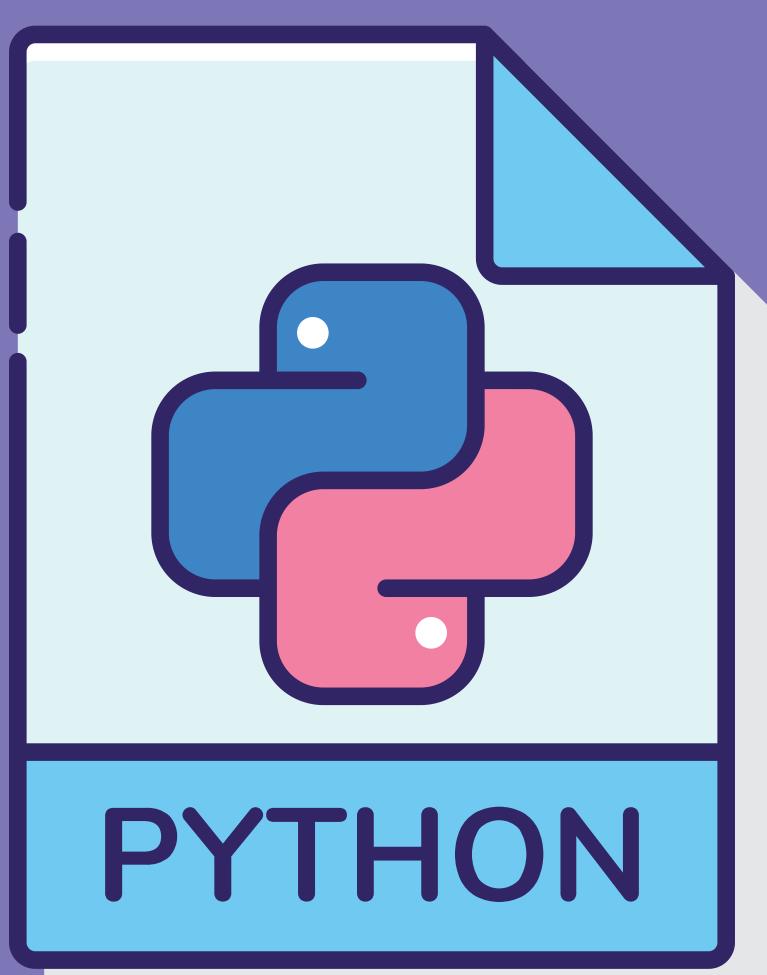


RIEPILOGO DI TUTTE LE COMPETENZE DI PYTHON SVOLTE NEL PRIMO QUADRIMESTRE



PYTHON: IL LINGUAGGIO DI PROGRAMMAZIONE

Python è un linguaggio di programmazione noto per la sua sintassi chiara e leggibile, rendendolo il linguaggio ideale per i principianti. Grazie alla sua versatilità, può essere utilizzato in una vasta gamma di applicazioni, dallo sviluppo web alla data science, dall'automazione di compiti quotidiani all'intelligenza artificiale. Python è anche noto per l'ecosistema ricco di librerie e frameworks che lo rendono uno strumento versatile e potente. Inoltre, Python è supportato da una comunità accogliente e inclusiva che aiuta gli sviluppatori a crescere professionalmente e a condividere le loro esperienze.



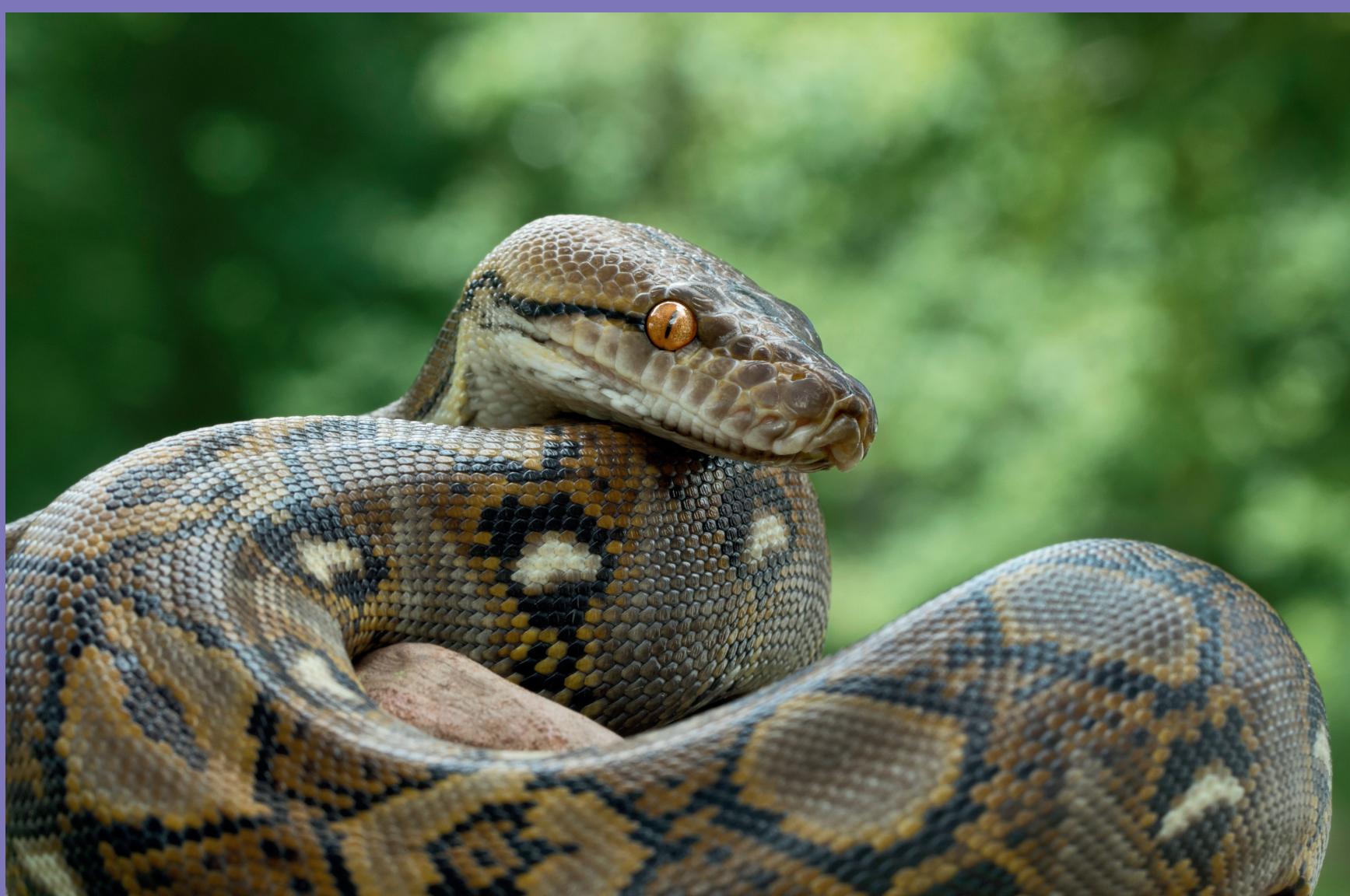
INDICE

Es. 1 (introduzione a Python, le basi): pag. 4

Es. 2 (matematica, fisica e giochi in Python): pag. 19

Es. 3 (i grafici): pag. 87

Es. 4 (i dataset): pag. 114



Es. 1 (introduzione a Python, le basi) (Matteo)

February 1, 2024

1 LE BASI DI PYTHON - IL MIO PRIMO PROGRAMMA

Questo breve codice utilizza il comando print per mostrare il messaggio “Hello, World!” a schermo. In Python, il print è il comando utilizzato per visualizzare testo o risultati durante l’esecuzione di un programma. Nel caso specifico, il programma stampa semplicemente un saluto di base. L’Hello World è il programma esempio “base” di un qualsiasi linguaggio di programmazione; infatti, come in questo caso, è il primo codice che si apprende.

```
[2]: #Comprendere il comando print  
      print("Hello, World!")#il comando print stampa un testo indicato nelle  
      ↪virgolette, quello spazio è inteso come una stringa di testo
```

Hello, World!

Questo codice utilizza il comando print per mostrare il contenuto della variabile nome. Il programma assegna la stringa “Matteo” alla variabile nome, la stringa non è altro che una frase di testo normale in linguaggio “umano” come è per l’appunto la parola “Matteo” che è una parola del linguaggio “umano” che viene assegnata come dato ad una variabile ed è una stringa. Successivamente, il programma utilizza il comando print per visualizzare il messaggio “Il nome esempio è:” seguito dal contenuto della variabile nome. Questo significa che verrà stampato a schermo “Il nome esempio è: Matteo”, ovviamente la parola “Matteo” è a capo perchè i due print sono in due righe diverse nel codice e questo fa sì che il risultato finale venga “printato” in due righe separate. In sintesi, il comando print può essere utilizzato anche per visualizzare il contenuto di variabili, come il nome nella variabile “nome”, o di qualsiasi altro dato.

```
[1]: #Comprendere il comando print e comprendere come definire una variabile con una  
      ↪stringa  
      nome="Matteo"  
      print("Il nome esempio è:")  
      print(nome)
```

Il nome esempio è:

Matteo

2 LE VARIABILI E L’INPUT UTENTE

Questo codice utilizza il comando input per acquisire dati dall’utente. Ecco una breve spiegazione di come funziona: Il programma per prima cosa chiede all’utente di inserire il proprio nome utilizzando

il comando input, l'input viene quindi memorizzato come un dato nella variabile nome. Successivamente, viene richiesto all'utente di inserire il cognome utilizzando l'altro input, quest'ultimo a quel punto viene sempre memorizzato come un dato nella variabile cognome. Infine, il programma stampa un messaggio di benvenuto utilizzando il comando print("Benvenuto su Jupyter Notebook", nome, cognome), dove nome e cognome sono i dati inseriti dall'utente poichè sono le variabili e quindi il programma stamperà direttamente i loro valori, cioè i dati che aveva immesso prima l'utente nell'input. Questo codice è un esempio semplice ma utile di come interagire con l'utente attraverso l'input e di come salvare i dati corretti nelle variabili.

```
[1]: #Comprendere il comando input
nome=input("Inserisci il tuo nome: ")#l'input "prende" un dato dall'utente e lo
    ↪salva temporaneamente nella RAM (Random Access Memory)
cognome=input("Inserisci il tuo cognome: ")#le variabili si possono indicare
    ↪come si vogliono però è meglio, per se stessi e per gli altri, che
    ↪visualizzeranno il codice usare dei nomi sensati per ogni occasione
print("Benvenuto su Jupyter Notebook",nome,cognome)#dentro il print ci sono due
    ↪input dettati dall'utente che specificano i dati
```

```
Inserisci il tuo nome: Matteo
Inserisci il tuo cognome: Magrino
Benvenuto su Jupyter Notebook Matteo Magrino
```

Il codice qui sotto chiede all'utente di inserire il nome completo della via e l'indirizzo di casa e successivamente stampa il messaggio confermando la via inserita. Questo codice è identico strutturalmente a quello di sopra perchè chiede sempre un'informazione all'utente via input, solo che questo è un esempio diverso

CURIOSITÀ: la via 20 W 34th St., New York, NY 10001, Stati Uniti è quella dove risiede una delle due facciate laterali lunghe del famosissimo grattacielo “Empire State Building”. Link alla pagina di Google Maps: <https://www.google.it/maps/place/Empire+State+Building/@40.7483306,-73.9861507,18.26z/data=!3m1!5s0x8b398fec1aea119:0x76fa1e3ac5a94c70!4m6!3m5!1s0x89c259a9b3117469:0xd13473.9856644!16zL20vMDJuZF8?entry=ttu>

```
[1]: indirizzoviadicasa=input("Inserisci il nome completo della via, nonchè
    ↪l'indirizzo, di casa tua: ")
print("Hai inserito la via:",indirizzoviadicasa)
```

```
Inserisci il nome completo della via, nonchè l'indirizzo, di casa tua: 20 W 34th
St., New York, NY 10001, Stati Uniti
Hai inserito la via: 20 W 34th St., New York, NY 10001, Stati Uniti
```

Il programma sottostante acquisisce il nome e il cognome dell'utente come input, così come il numero di volte che l'utente desidera ripetere un messaggio gentile. Utilizza poi un ciclo for per iterare attraverso il numero di volte specificato dall'utente, gestito dal contatore “i” (che sta per iteratore ma si può chiamare come si vuole). In ogni iterazione del ciclo, il programma stampa un messaggio gentile che include il nome e il cognome inseriti dall'utente. La parte chiave del programma è ovviamente il ciclo for, che utilizza un contatore o iteratore (i due nomi sono sinonimi) “i” per iterare attraverso un numero di volte pari a quello specificato dalla variabile numeroripetizionimessaggio. In ogni iterazione, il programma stampa un messaggio gentile che include il nome e il cognome inseriti dall'utente. Il contatore “i” è un vero e proprio iteratore, che

tiene traccia dello stato attuale del ciclo e consente di eseguire azioni specifiche per ciascuna iterazione. La parte del range(numeroripetizionimessaggio) fornisce una sequenza di numeri da 0 a numeroripetizionimessaggio-1, definendo così il numero di iterazioni.

3 IL CICLO FOR

```
[1]: #Comprendere il ciclo for
nome=input("Inserisci il tuo nome: ")
cognome=input("Inserisci il tuo cognome: ")
numeroripetizionimessaggio=int(input("Inserisci il numero di volte che desideri che un messaggio gentile si ripeta: "))#int è l'acronimo di integer, cioè di intero e vuole dire che la variabile in questo caso è di tipo intero e può leggere solo numeri interi (cioè tutti quelli senza virgola) o altro
#For è un contatore o iteratore perchè tiene traccia dello stato attuale del ciclo (per spiegazione più dettagliata su come funziona leggere il riquadro sopra)
for i in range(numeroripetizionimessaggio):#il numero di volte che viene ripetuto questo messaggio dipende dall'input messo precedentemente dall'utente
    print("Che bel nome e cognome che hai",nome,cognome,"è davvero molto bello")
```

```
Inserisci il tuo nome: Matteo
Inserisci il tuo cognome: Magrino
Inserisci il numero di volte che desideri che un messaggio gentile si ripeta: 12
Che bel nome e cognome che hai Matteo Magrino è davvero molto bello
Che bel nome e cognome che hai Matteo Magrino è davvero molto bello
Che bel nome e cognome che hai Matteo Magrino è davvero molto bello
Che bel nome e cognome che hai Matteo Magrino è davvero molto bello
Che bel nome e cognome che hai Matteo Magrino è davvero molto bello
Che bel nome e cognome che hai Matteo Magrino è davvero molto bello
Che bel nome e cognome che hai Matteo Magrino è davvero molto bello
Che bel nome e cognome che hai Matteo Magrino è davvero molto bello
Che bel nome e cognome che hai Matteo Magrino è davvero molto bello
Che bel nome e cognome che hai Matteo Magrino è davvero molto bello
Che bel nome e cognome che hai Matteo Magrino è davvero molto bello
Che bel nome e cognome che hai Matteo Magrino è davvero molto bello
```

4 ESEMPI DI CALCOLATRICE

Il codice inizia con un messaggio di benvenuto, invitando l'utente a utilizzare la calcolatrice per effettuare addizioni. Successivamente, attraverso il comando input, si richiede all'utente di inserire due numeri, che vengono poi convertiti in formato float per consentire la lettura da parte del programma e l'immisione di numeri decimali da parte dell'utente. L'operazione di addizione tra i due numeri viene eseguita utilizzando l'operatore + (più), il risultato viene memorizzato nella variabile addizione, e infine, il risultato viene visualizzato con un messaggio appropriato. **N.B.:** la virgola nei numeri decimali in Python quando vengono immessi come input, come nella maggior parte dei linguaggi, DEVE sempre essere inserita con un punto e non con una virgola questo perchè per convenzione si è scelto di assegnare il punto come

separatore decimale.

```
[3]: #Comprendere come eseguire le addizioni
print("Ciao, attraverso questa calcolatrice puoi eseguire calcoli con le addizioni!")
numero1=float(input("Inserisci il primo numero: "))#float indica che la variabile può leggere i numeri reali, cioè tutti quei numeri inclusi quelli con la virgola o senza
numero2=float(input("Inserisci il secondo numero: "))
addizione=numero1+numero2
print("La somma dei due numeri è pari a:",float(addizione))
```

Ciao, attraverso questa calcolatrice puoi eseguire calcoli con le addizioni!
Inserisci il primo numero: 5.7
Inserisci il secondo numero: 3.2
La somma dei due numeri è pari a: 8.9

Il codice qui sotto svolge l'operazione di sottrazione tra due numeri (sempre float) inseriti dall'utente. Funziona allo stesso modo ed infatti ha una identica struttura tranne per il fatto che c'è il simbolo - (meno) che fa eseguire correttamente l'operazione in questione.

```
[2]: #Comprendere come eseguire le sottrazioni
print("Ciao, attraverso questa calcolatrice puoi eseguire calcoli con le sottrazioni!")
numero1=float(input("Inserisci il primo numero: "))
numero2=float(input("Inserisci il secondo numero: "))
sottrazione=numero1-numero2
print("La sottrazione dei due numeri è pari a:",float(sottrazione))
```

Ciao, attraverso questa calcolatrice puoi eseguire calcoli con le sottrazioni!
Inserisci il primo numero: 3.4
Inserisci il secondo numero: 8.2
La sottrazione dei due numeri è pari a: -4.799999999999999

Il codice qui sotto svolge l'operazione di moltiplicazione tra due numeri (sempre float) inseriti dall'utente. Funziona allo stesso modo ed infatti ha una identica struttura tranne per il fatto che c'è il simbolo * (asterisco) che in programmazione, non solo in Python ma anche nella maggior parte dei linguaggi, vuole significare il simbolo per (x) che fa eseguire correttamente l'operazione in questione.

```
[3]: #Comprendere come eseguire le moltiplicazioni
print("Ciao, attraverso questa calcolatrice puoi eseguire calcoli con le moltiplicazioni!")
numero1=float(input("Inserisci il primo numero: "))
numero2=float(input("Inserisci il secondo numero: "))
moltiplicazione=numero1*numero2#il simbolo * vuol dire moltiplicazione
print("La moltiplicazione dei due numeri è pari a:",float(moltiplicazione))
```

Ciao, attraverso questa calcolatrice puoi eseguire calcoli con le moltiplicazioni!
Inserisci il primo numero: 3.9

```
Inserisci il secondo numero: 9.1
La moltiplicazione dei due numeri è pari a: 35.489999999999995
```

Il codice qui sotto svolge l'operazione di divisione tra due numeri (sempre float) inseriti dall'utente. Funziona allo stesso modo ed infatti ha una identica struttura tranne per il fatto che c'è il simbolo / (barra o slash in inglese) che in programmazione, non solo in Python ma anche nella maggiorparte dei linguaggi, vuole significare il simbolo della divisione (cioè il diviso) (: oppure il \div) che fa eseguire correttamente l'operazione in questione. Anche se si può dire che in algebra o semplicemente in matematica il simbolo / viene usato per scrivere le frazioni, che sono comunque delle divisioni a tutti gli effetti.

```
[1]: #comprendere come eseguire le divisioni
print("Ciao, attraverso questa calcolatrice puoi eseguire calcoli con le divisioni!")
numero1=float(input("Inserisci il primo numero: "))
numero2=float(input("Inserisci il secondo numero: "))
divisione=numero1/numero2#il simbolo / vuol dire divisione
print("La divisione dei due numeri è pari a:",float(divisione))
```

```
Ciao, attraverso questa calcolatrice puoi eseguire calcoli con le divisioni!
Inserisci il primo numero: 5.7
Inserisci il secondo numero: 2.6
La divisione dei due numeri è pari a: 2.1923076923076925
```

Il codice qui sotto svolge l'operazione di potenza tra due numeri (la base come float e l'esponente come int) inseriti dall'utente. Funziona allo stesso modo ed infatti ha una identica struttura tranne per il fatto che c'è il simbolo ** (doppio asterisco, che letteralmente vorrebbe dire doppio \times) che in programmazione, non solo in Python ma anche nella maggiorparte dei linguaggi, vuole significare il simbolo della potenza (x^y , come il ** se fosse il \wedge) che fa eseguire correttamente l'operazione in questione.

```
[5]: #Comprendere come eseguire le potenze
print("Ciao, attraverso questa calcolatrice puoi calcolare il valore di una potenza!")
base=float(input("Inserisci la base della potenza: "))#la base è un float, questo perchè si può fare la potenza di un numero decimale
esponente=int(input("Inserisci l'esponente della potenza: "))#l'esponente è un int invece, questo perchè per fare un esempio più facile è meglio usare l'int in modo che il risultato sia più semplice
potenza=base**esponente#il simbolo ** vuol dire potenza
print("Il valore della potenza è pari a:",float(potenza))
```

```
Ciao, attraverso questa calcolatrice puoi calcolare il valore di una potenza!
Inserisci la base della potenza: 5.7
Inserisci l'esponente della potenza: 3
Il valore della potenza è pari a: 185.193
```

Questo codice riassume tutti i precedenti in un'unico codice. Il codice infatti rappresenta una calcolatrice che consente all'utente di eseguire operazioni di addizione, sottrazione, moltiplicazione, divisione e calcolo delle potenze. Questa versatilità è ottenuta attraverso l'utilizzo di condizioni (che

in ordine gerarchico sono: if, elif ed else) che dirigono il flusso del programma in base all'operazione scelta dall'utente. Inizialmente, il programma richiede all'utente di inserire il tipo di operazione che desidera effettuare sottoforma di segno, nonché poi i due numeri coinvolti. L'input dell'utente viene successivamente convertito in numeri in virgola decimale (i cosiddetti float) per consentire l'elaborazione di numeri decimali. La parte principale del codice è però costituita dalle condizioni if, elif ed else (in scala gerarchica). Ciascuna condizione verifica se l'operatore scritto nell'input corrisponde a uno specifico caso dichiarato in precedenza (addizione, sottrazione, moltiplicazione, divisione, potenza) e, in caso affermativo, esegue il blocco di codice associato a quella condizione, cioè il blocco di codice scritto sotto alla condizione finché non incomincia una nuova condizione perché poi a quel punto il codice salta quel pezzo delle condizioni e va alla parte dopo. Se l'operazione inserita non corrisponde a nessun caso noto, il blocco nell'else fornisce un messaggio di errore. Inoltre l'else è sempre l'ultima condizione mentre la prima è sempre l'if e quelle in mezzo sono sempre l'elif. Questo approccio rende il codice estremamente flessibile, consentendo allo sviluppatore di eseguire diverse operazioni senza dover scrivere del codice separato per ciascuna di esse, utilizzando per l'appunto le condizioni. Le condizioni giocano un ruolo fondamentale nell'indirizzare il flusso del programma in modo dinamico, in base alle scelte dell'utente, offrendo così in questo caso una calcolatrice completa e versatile.

```
[1]: #Esempio di calcolatrice finale con tutte cinque le operazioni
print("Ciao, attraverso questa calcolatrice puoi eseguire calcoli con le
      →addizioni, sottrazioni, moltiplicazioni e divisioni, inoltre puoi anche
      →calcolare il valore di una potenza!")

operazione=input("Adesso scegli l'operazione che fa al caso tuo attraverso il
      →correlato simbolo (+, -, *, /, **): ")

numero1=float(input("Inserisci il primo numero o la base della potenza: "))
numero2=float(input("Inserisci il secondo numero o l'esponente della potenza: "))

if operazione=="+":#l'if indica il primo caso possibile
    risultato=numero1+numero2
elif operazione=="-":##l'elif è l'ibrido tra if & else e indica i casi di mezzo
    risultato=numero1-numero2
elif operazione=="*":
    risultato=numero1*numero2
elif operazione=="/":
    risultato=numero1/numero2
elif operazione=="**":
    risultato=numero1**numero2
else:#l'else indica l'ultimo caso possibile
    risultato="operatore non riconosciuto. Scegli uno dei seguenti operatori
      →riavviando prima il programma: +, -, *, /, **"
print("Il risultato finale del calcolo è:",risultato)
```

Ciao, attraverso questa calcolatrice puoi eseguire calcoli con le addizioni, sottrazioni, moltiplicazioni e divisioni, inoltre puoi anche calcolare il valore di una potenza!

Adesso scegli l'operazione che fa al caso tuo attraverso il correlato simbolo (+, -, *, /, **): non lo so

Inserisci il primo numero o la base della potenza: 5

Inserisci il secondo numero o l'esponente della potenza: 5

Il risultato finale del calcolo è: operatore non riconosciuto. Scegli uno dei seguenti operatori riavviando prima il programma: +, -, *, /, **

5 IL CICLO FOR CON N

Il programma in questione è un semplice “generatore” di numeri che parte e arriva fino a un numero desiderato dall’utente. Inizia chiedendo all’utente due numeri positivi: il primo (numerodaverificare) indica fino a quale numero si vuole generare la lista della sequenza di numeri, mentre il secondo (numeroiniziale) rappresenta da quale numero positivo si desidera iniziare la numerazione. Successivamente, attraverso un ciclo for, il programma itera attraverso una sequenza di numeri a partire da “numeroiniziale” fino a “numerodaverificare” incluso. Durante ogni iterazione del ciclo, il numero corrente viene stampato a schermo dopo una stringa esplicativa ed è così per ogni volta che si ripete questo ciclo. Questo processo continua fino a quando il ciclo raggiunge il valore scelto dall’utente della variabile “numerodaverificare”. Quindi alla fine si può dire che l’uso di range(numeroiniziale, numerodaverificare + 1) assicura che il valore finale “numeroiniziale” sia incluso nella sequenza, poiché senza il +1, la sequenza generata terminerebbe a “numeroiniziale” - 1, cioè al numero precedente da quello scelto dall’utente, quindi escludendo il valore desiderato “numerodaverificare” dalla lista finale che viene stampata. In questo modo, il +1 garantisce che la numerazione vada da “numeroiniziale” fino a “numerodaverificare” scrivendo tutti i numeri della lista, inclusi entrambi i “poli” decisi come input dall’utente. Questo tipo di programma è utile quando si desidera generare e visualizzare una sequenza ordinata di numeri in base, però, alle preferenze dell’utente.

```
[5]: #Comprendere il ciclo for con n
print("Ciao, attraverso questo programma puoi verificare la numerazione ordinaria dei numeri dal e fino al punto che desideri")
numerodaverificare=int(input("Quindi, inserisci il numero positivo che desideri verificare: "))
numeroiniziale=int(input("Poi, inserisci da quale numero positivo desideri che la numerazione inizii: "))
print("La numerazione ordinaria dei numeri dal e fino al punto che desideri è:")
for numero in range(numeroiniziale,numerodaverificare+1):#p è un valore fisso determinato dall'utente mentre +1 indica che a n verrà aggiunto il valore di 1.
    ↪N.B.: la variabile numero è diversa dalle altre due infatti rappresenta ciascun numero nella numerazione ordinaria tra numeroiniziale e numerodaverificare, cioè assume successivamente ogni valore compreso tra numeroiniziale e numerodaverificare e non ce mai bisogno di inizzalizzarla in Python
    print(numero)
```

Ciao, attraverso questo programma puoi verificare la numerazione ordinaria dei numeri dal e fino al punto che desideri
Quindi, inserisci il numero positivo che desideri verificare: 33
Poi, inserisci da quale numero positivo desideri che la numerazione inizii: 12
La numerazione ordinaria dei numeri dal e fino al punto che desideri è:
12
13
14
15

16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33

Il programma sottostante calcola la somma cumulativa dei primi n numeri interi positivi chiedendo il numero desiderato all'utente come input. La somma cumulativa è un concetto matematico che rappresenta la somma progressiva di una sequenza di numeri, aggiungendo ad ogni passo il numero corrente all'accumulatore. In questo contesto, il programma calcola la somma cumulativa dei primi n numeri interi positivi, fornendo un risultato che riflette l'incremento graduale della somma durante l'iterazione. Ad esempio, se l'utente inserisce n=5, la somma cumulativa sarà calcolata come $1+2+3+4+5$, che darà come risultato 15. Il programma inizia chiedendo all'utente di inserire come input un numero intero positivo (variabile n) di cui desidera calcolare la somma cumulativa. Successivamente, il programma utilizza un ciclo for per iterare attraverso i numeri da 1 a n inclusi e quindi dal numero 1 fino al numero scelto dall'utente. Durante ogni iterazione, il programma aggiunge il numero corrente alla variabile addizione, che funge da accumulatore per la somma cumulativa, finché il processo di iterazione non sarà arrivato al numero scelto dall'utente. L'operatore += usato in questo codice ad ogni ciclo di iterazione esegue contemporaneamente l'addizione e la riassegnazione del risultato alla variabile addizione, cioè per prima cosa fa il calcolo dell'addizione che deve svolgere e poi riassegna il valore del risultato di quel calcolo alla variabile addizione e questo permette di far due azione che dovrebbero essere assegnate con due comandi diversi in uno singolo in modo da risparmiare memoria. Infine, il programma stampa il risultato, indicando la somma dei primi n numeri interi e sottolineando che questo valore rappresenta la somma cumulativa. Questo codice fornisce un'implementazione semplice della somma cumulativa.

[3]: #Comprendere il ciclo for con n facendo la somma cumulativa
print("Ciao, attraverso questo programma puoi calcolare la somma cumulativa di
→qualsiasi numero che desideri sapere")
n=int(input("Quindi, inserisci il numero positivo che desideri calcolare: "))
addizione=0
for numero in range(1,n+1):#1 è un valore fisso mentre +1 indica che a n verrà
→aggiunto il valore di 1
 addizione+=numero#il simbolo += svolge due compiti in uno: fa la somma e
→riassegna il valore della variabile

```

print("La somma dei primi",n,"numeri interi è",addizione)
print("Perciò la somma cumulativa di",n,"è",addizione)

```

Ciao, attraverso questo programma puoi calcolare la somma cumulativa di qualsiasi numero che desideri sapere
 Quindi, inserisci il numero positivo che desideri calcolare: 12
 La somma dei primi 12 numeri interi è 78
 Perciò la somma cumulativa di 12 è 78

6 CALCOLARE LA MEDIA DI UNA LISTA DI NUMERI

Questo programma consente all'utente di calcolare la media di una lista di numeri scelta dall'utente stesso. La media dei numeri è ottenuta sommando tutti i numeri insieme e dividendo il risultato per il numero totale di numeri. Inizialmente, il programma chiede all'utente quanti numeri desidera inserire nella lista. Utilizzando poi un ciclo for, il programma acquisisce ogni numero specifico della lista inserito dall'utente e li aggiunge alla lista numeri utilizzando il metodo append, che per l'appunto "appende" ognuno dei numeri scritti all'utente e salvati nella variabile "numero" alla lista vera e propria del programma. Successivamente, viene calcolata la media dei numeri presenti nella lista utilizzando i comandi sum e len. Il comando sum(numeri) restituisce la somma di tutti i numeri nella lista, mentre len(numeri) restituisce la lunghezza della lista, cioè il numero totale di numeri inseriti. Dividendo la somma per la lunghezza, si ottiene così la media dei numeri. Infine il programma stampa la lista finale, fornendo così all'utente la media di tutti i numeri inseriti da lui. Questo codice fornisce un modo efficiente per calcolare la media di una lista di numeri inseriti dall'utente.

```

[1]: #Comprendere come usare i comandi sum e len facendo la media di una lista di numeri
      #l'append serve a fare una lista da poter richiamare
      #Ciao, attraverso questo programma puoi calcolare la media di una lista di numeri che desideri sapere"
      n=int(input("Quanti numeri desideri inserire in totale? "))
      for i in range(n):
          numero=float(input("Inserisci il numero di cui desideri fare la media: "))
          numeri.append(numero)#così si crea la lista che varia dalla variabile numero, "appendendo" il numero scritto dall'utente prima alla lista vera e propria
      mediadeinumeri=sum(numeri)/len(numeri)#sum=somma mentre len=lunghezza
      print("La media di tutti i numeri inseriti è:",mediadeinumeri)

```

Ciao, attraverso questo programma puoi calcolare la media di una lista di numeri che desideri sapere
 Quanti numeri desideri inserire in totale? 5
 Inserisci il numero di cui desideri fare la media: 12
 Inserisci il numero di cui desideri fare la media: 33
 Inserisci il numero di cui desideri fare la media: 57
 Inserisci il numero di cui desideri fare la media: 127
 Inserisci il numero di cui desideri fare la media: 73
 La media di tutti i numeri inseriti è: 60.4

7 CALCOLARE IL QUADRATO DEI NUMERI

Il programma consente all'utente di calcolare il quadrato di un numero specificato sempre dall'utente. Il quadrato di un numero è ottenuto moltiplicando il numero per se stesso, per l'appunto facendo la potenza alla seconda del numero in questione. Inizialmente, il programma richiede all'utente di inserire come input un numero intero (variabile n) di cui desidera calcolare il suo quadrato. Successivamente, attraverso un ciclo for, il programma itera attraverso i numeri da 1 a n inclusi. Per ogni numero da 1 a n, il ciclo for calcola il quadrato utilizzando l'operatore di potenza `**` e poi stampa il risultato dopo una stringa di testo esplicativa. Il risultato finale di output è una lista di quadrati dei primi n numeri, con ogni riga che mostra all'utente il numero originale e il suo quadrato corrispondente. Questo codice fornisce un modo semplice per calcolare il quadrato di un numero.

```
[1]: #Comprendere come eseguire le potenze facendo il quadrato dei numeri
print("Ciao, attraverso questo programma puoi calcolare il quadrato di un numero che
      →che desideri sapere")
n=int(input("Quindi, inserisci il numero di cui desideri sapere il quadrato: "))
print("I quadrati dei primi",n,"numeri sono: ")
for numero in range(1,n+1):
    quadratodelnumero=numero**2#il simbolo di potenza si fa con il doppio **
    print("Il quadrato di",numero,"è",quadratodelnumero)
```

Ciao, attraverso questo programma puoi calcolare il quadrato di un numero che desideri sapere

Quindi, inserisci il numero di cui desideri sapere il quadrato: 12

I quadrati dei primi 12 numeri sono:

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
Il quadrato di 9 è 81
Il quadrato di 10 è 100
Il quadrato di 11 è 121
Il quadrato di 12 è 144

8 LA VERIFICA DI PARITÀ O DISPARITÀ DI UN NUMERO

Il programma consente all'utente di verificare se un numero inserito come input è pari o dispari. Inizialmente, chiede all'utente di inserire un numero intero come input della variabile numero. Successivamente, utilizza l'operatore modulo “%” (simbolo di percentuale) per verificare se il numero è divisibile per 2 o meno. Se il resto della divisione di numero per 2 è uguale a 0, il programma stampa che il numero è pari. In caso contrario, se il resto è diverso da 0, il programma stampa che il numero è dispari. L'operatore modulo restituisce il resto della divisione tra i due operandi, cioè in questo caso il numero definito dall'utente e il 2. Nello specifico nella condizione “if numero%2==0:”, il programma verifica se il numero è divisibile per 2 (cioè se il resto è 0), il che indica che il numero

è pari. L'else ovviamente, se il programma si trova nella altra condizione, stampa che è dispari. Questo codice fornisce una semplice verifica della parità o disparità di un numero.

```
[2]: #Comprendere l'utilizzo del simbolo di %
print("Ciao, attraverso questo programma puoi verificare se un numero è pari o_
→dispari")
numero=int(input("Quindi, inserisci il numero che desideri verificare: "))
if numero%2==0:#il simbolo di % vuol dire se è divisibile o meno un numero, ==0_
→vuol dire se in quel caso è con il resto di 0
    print("Il numero",numero,"è un numero pari")
else:
    print("Il numero",numero,"è un numero dispari")
```

Ciao, attraverso questo programma puoi verificare se un numero è pari o dispari
Quindi, inserisci il numero che desideri verificare: 12
Il numero 12 è un numero pari

9 CALCOLARE IL FATTORIALE

Il programma qui sotto consente all'utente di calcolare il fattoriale di un numero specificato dall'utente stesso. Il fattoriale di un numero è ottenuto moltiplicando tutti i numeri interi positivi fino a quel numero, il fattoriale di 5 ad esempio è calcolato moltiplicando tutti i numeri interi positivi da 1 a 5, cioè: $1 * 2 * 3 * 4 * 5$, che è uguale a 120. Inizialmente, il programma richiede all'utente come input di inserire un numero positivo (variabile n) di cui desidera calcolare il fattoriale. Successivamente, attraverso un ciclo for, il programma itera attraverso i numeri da 1 a n inclusi. Per ogni numero, il ciclo for moltiplica il valore corrente della variabile fattoriale per il numero stesso e riassegna il risultato alla variabile fattoriale. L'operatore = è un modo conciso e semplice per eseguire moltiplicazioni e riassegnare il risultato alla stessa variabile, praticamente fa due cose in uno: svolge la moltiplicazione e salva il risultato nella stessa variabile senza l'uso di due comandi separati. Ad esempio, fattoriale=numero è equivalente a fattoriale=fattoriale*numero solo che per quest'ultimo il programma userà più memoria. Alla fine del ciclo, dopo una stringa esplicativa, il programma stampa il risultato che rappresenta il fattoriale del numero inserito all'inizio dall'utente.

```
[1]: #Comprendere l'utilizzo del simbolo di *=
print("Ciao, attraverso questo programma puoi calcolare il fattoriale di_
→qualsiasi numero che desideri sapere")
n=int(input("Quindi, inserisci il numero positivo che desideri calcolare: "))
fattoriale=1
for numero in range(1,n+1):
    fattoriale*=numero#il simbolo *= svolge due compiti in uno: fa la_
→moltiplicazione e riassegna il valore della variabile
print("Il fattoriale del numero",n,"è",fattoriale)
```

Ciao, attraverso questo programma puoi calcolare il fattoriale di qualsiasi numero che desideri sapere
Quindi, inserisci il numero positivo che desideri calcolare: 12
Il fattoriale del numero 12 è 479001600

10 IL GIOCO DELL'INDOVINELLO

Il programma in questione è un semplice gioco in cui l'utente, o in questo caso il giocatore, deve indovinare un numero generato casualmente compreso tra 1 e 100. Inizialmente, il programma utilizza la libreria random per generare un numero segreto (salvato nella variabile numerodaindovinare) mediante la funzione randint(1, 100), che restituisce un numero intero casuale compreso tra 1 e 100 (inclusi). Successivamente, il programma inizia un ciclo “while True” che si ripeterà finché l'utente non indovina il numero segreto. Il ciclo “while True” non è altro che un ciclo che dura fino all’infinito, cioè fino a quando l'utente non si sarà imbattuto nell'unica condizione del programma che attraverso il comando “break” riuscirà ad interrompere questo ciclo continuo e far così terminare il programma; se durante lo sviluppo del programma ci si dimentica di inserire questo comando il programma potrebbe realmente andare continuare “per sempre”, anche se questo sarebbe scientificamente impossibile. Durante ogni iterazione del ciclo, il programma chiede all'utente di inserire un tentativo numerico (salvato nella variabile tentativo). Il programma tiene traccia del numero di tentativi effettuati con la variabile “tentativi”. Se l'utente riesce ad indovinare il numero segreto, il programma stampa un messaggio di congratulazioni insieme al numero segreto stampato a schermo e al numero di tentativi effettuati, quindi a quel punto quest'ultimo esce dal ciclo con l'istruzione “break”. Se il tentativo dell'utente è sbagliato, il programma fornisce un suggerimento sulla direzione corretta per aiutare il giocatore (in particolare il programma specifica se il numero è più grande o più piccolo, questo grazie a delle condizioni specifiche inserite nel programma).

```
[3]: #Comprendere l'utilizzo della libreria random
import random
numerodaindovinare=random.randint(1,100)#random.randint vuol dire che il numero
→da generare è compreso da 1 a 100 ed è intero perché è int
tentativi=0
print("Ciao, questo gioco prevede che tu indovini un numero compreso da 1 a 100")
while True:
    tentativo=int(input("Adesso inserisci il numero che ritieni sia quello"))
    →giusto: " ")
    tentativi += 1
    if tentativo==numerodaindovinare:
        print("Complimenti! Sei riuscito a indovinare il numero incognito che"
→è,numerodaindovinare,"in",tentativi,"tentativi")
        break#fa fermare "improvvisamente" il programma
    elif tentativo<numerodaindovinare:
        print("Ci sei quasi, ti stai avvicinando! Ti do un consiglio: il numero"
→è più grande)
    else:
        print("Ci sei quasi, ti stai avvicinando! Ti do un consiglio: il numero"
→è più piccolo")
```

Ciao, questo gioco prevede che tu indovini un numero compreso da 1 a 100
Adesso inserisci il numero che ritieni sia quello giusto: 10
Ci sei quasi, ti stai avvicinando! Ti do un consiglio: il numero è più grande
Adesso inserisci il numero che ritieni sia quello giusto: 90
Ci sei quasi, ti stai avvicinando! Ti do un consiglio: il numero è più piccolo
Adesso inserisci il numero che ritieni sia quello giusto: 20

Ci sei quasi, ti stai avvicinando! Ti do un consiglio: il numero è più grande
 Adesso inserisci il numero che ritieni sia quello giusto: 80
 Ci sei quasi, ti stai avvicinando! Ti do un consiglio: il numero è più piccolo
 Adesso inserisci il numero che ritieni sia quello giusto: 30
 Ci sei quasi, ti stai avvicinando! Ti do un consiglio: il numero è più grande
 Adesso inserisci il numero che ritieni sia quello giusto: 70
 Ci sei quasi, ti stai avvicinando! Ti do un consiglio: il numero è più piccolo
 Adesso inserisci il numero che ritieni sia quello giusto: 40
 Ci sei quasi, ti stai avvicinando! Ti do un consiglio: il numero è più grande
 Adesso inserisci il numero che ritieni sia quello giusto: 60
 Ci sei quasi, ti stai avvicinando! Ti do un consiglio: il numero è più grande
 Adesso inserisci il numero che ritieni sia quello giusto: 61
 Ci sei quasi, ti stai avvicinando! Ti do un consiglio: il numero è più grande
 Adesso inserisci il numero che ritieni sia quello giusto: 62
 Ci sei quasi, ti stai avvicinando! Ti do un consiglio: il numero è più grande
 Adesso inserisci il numero che ritieni sia quello giusto: 63
 Ci sei quasi, ti stai avvicinando! Ti do un consiglio: il numero è più grande
 Adesso inserisci il numero che ritieni sia quello giusto: 64
 Ci sei quasi, ti stai avvicinando! Ti do un consiglio: il numero è più grande
 Adesso inserisci il numero che ritieni sia quello giusto: 65
 Ci sei quasi, ti stai avvicinando! Ti do un consiglio: il numero è più grande
 Adesso inserisci il numero che ritieni sia quello giusto: 67
 Ci sei quasi, ti stai avvicinando! Ti do un consiglio: il numero è più grande
 Adesso inserisci il numero che ritieni sia quello giusto: 68
 Complimenti! Sei riuscito a indovinare il numero incognito che è 68 in 15 tentativi

Qui sotto è presente un breve codice che mostra come generare un numero causale intero (int), questa parte di codice è anche presente nel codice soprastante ma ovviamente con variabili diverse.

```
[19]: #Comprendere l'utilizzo della libreria random per generare numeri casuali interi
import random
print("Un numero di esempio intero (int) generato casualmente da un range da 1 a 100 è: ")
numerointerocasuale=random.randint(1,100)#random.randint vuol dire che il numero da generare è compreso da 1 a 100 ed è intero perchè è int
print(numerointerocasuale)
```

Un numero di esempio intero (int) generato casualmente da un range da 1 a 100 è:
 27

Invece qui sotto è presente il metodo per farlo con i numeri decimali (float).

```
[22]: #Comprendere l'utilizzo della libreria random per generare numeri casuali decimali
import random
print("Un numero di esempio decimale (float) generato casualmente da un range da 0,12 a 99,7 è: ")
```

```

numerofloatcasuale=random.uniform(0.12, 99.7)#random.uniform vuol dire che il
→numero da generare è compreso da 0,12 a 99,07 ed è decimale perché è float
print(numerofloatcasuale)

```

Un numero di esempio decimale (float) generato casualmente da un range da 0,12 a 99,7 è:
86.19652177433156

11 IL GIOCO DEL MORRA CINESE

Nel codice del gioco della morra cinese, viene sfruttata la libreria random di Python per creare un gioco interattivo e coinvolgente. Inizialmente, il programma genera casualmente la scelta da parte del computer tra carta, forbici e sasso, consentendo così una migliore dinamicità e casualità nel gioco. Successivamente, l'utente viene coinvolto con un'interfaccia di input, in cui è richiesto di selezionare (cioè scrivere nella apposita casella di testo) la propria mossa preferita tra le opzioni disponibili: carta, forbici o sasso. Il programma fornisce poi all'utente la mossa scelta dal computer stampandola a schermo appena l'utente avrà scritto la sua mossa, creando così un'atmosfera di suspense. Le regole del gioco sono implementate attraverso delle condizioni logiche che confrontano le mosse dell'utente e del computer con delle scelte prefissate. Gli operatori logici and e or vengono utilizzati per definire le situazioni in cui l'utente può vincere, perdere o ottenere una parità contro il computer e vengono usate per poter specificare più cose contemporaneamente senza dover creare tantissime condizioni nel codice. Inoltre, il codice gestisce situazioni in cui l'utente inserisce una mossa non riconosciuta rispetto all'elenco delle opzioni scritte sopra, fornendo un messaggio di avviso e suggerendo di scegliere tra le opzioni valide riavviando però prima il programma. Un elemento interessante del codice è l'inclusione di un link esterno, in questo caso un link di Wikipedia, che invita l'utente a consultare ulteriori informazioni sul regolamento e le strategie del gioco della morra cinese. Complessivamente, il codice combina efficacemente l'input utente, logica condizionale e interazione dinamica per creare un'esperienza di gioco coinvolgente. Questo codice riprende quasi tutte le competenze raccolte finora in questa esercitazione.

[27]: #Riassunto di tutte le competenze imparate finora in questa esercitazione +
→comprendere l'utilizzo dei comandi not in, and e or

```

import random
mosse=["carta","forbici","sasso"]
mossadelcomputer=random.choice(mosse)#random.choice vuol dire che il programma
→sceglierà casualmente un elemento tra quelli della lista o stringa
print("Ciao, questo gioco è la morra cinese quindi devi provare ad indovinare la")
→mossa giusta e battere il computer che è l'avversario")
print("Puoi trovare tutto il regolamento e le strategie del gioco a questo link:")
→https://it.wikipedia.org/wiki/Morra_cinese")
sceltadelgiocatore=input("Quindi adesso scegli la tua mossa scrivendo una delle")
→seguenti opzioni: carta, forbice o sasso: ")
if sceltadelgiocatore not in mosse:#il comando not in viene utilizzato per
→verificare se un elemento NON è presente in una lista o stringa
    print("Mossa non riconosciuta. Scegli una delle seguenti mosse riavviando")
→prima il programma: carta, forbice o sasso")
    print("Impossibile perciò decretare un vincitore")

```

```

else:#all'interno di un else, o in una qualsiasi altra condizione, ci possono
→essere altre condizioni che si possono verificare, con altre condizioni a loro
→volta all'interno e così via dicendo
    print("Il computer ha scelto la seguente mossa:",mossadelcomputer)
    if sceltadelgiocatore==mossadelcomputer:
        print("Parità!")
    elif (sceltadelgiocatore=="carta" and
→mossadelcomputer=="sasso")or(sceltadelgiocatore=="forbici" and
→mossadelcomputer=="carta")or(sceltadelgiocatore=="sasso" and
→mossadelcomputer=="forbici"):#l'and serve a comparare più cose
→contemporaneamente mentre l'or funge come un oppure
        print("Complimenti! Hai vinto!")
    else:
        print("Peccato! Hai perso!")

```

Ciao, questo gioco è la morra cinese quindi devi provare ad indovinare la mossa giusta e battere il computer che è l'avversario

Puoi trovare tutto il regolamento e le strategie del gioco a questo link:

https://it.wikipedia.org/wiki/Morra_cinese

Quindi adesso scegli la tua mossa scrivendo una delle seguenti opzioni: carta, forbice o sasso: forbici

Il computer ha scelto la seguente mossa: carta

Complimenti! Hai vinto!

Es. 2 (matematica, fisica e giochi in Python) (Luca)

February 1, 2024

1 IL CALCOLO DEL FATTORIALE DI UN NUMERO

In questa modalità, il programma consente all'utente di calcolare la fattoriale di un numero intero positivo inserito e visualizzare il risultato.

```
[33]: #Chiedere all'utente di inserire un numero intero positivo
print("Ciao, attraverso questo programma puoi calcolare il fattoriale di un qualsiasi numero che desideri sapere")
numero=int(input("Quindi, inserisci il numero intero positivo che desideri calcolare: "))
fattoriale=1
if numero<0:
    print("Non è possibile calcolare il fattoriale di un numero negativo ")
elif numero==0:
    print("Il fattoriale di 0 è sempre 1")
else:
    for numero in range(1,numero+1):
        fattoriale*=numero#il simbolo *= svolge due compiti in uno: fa la somma e riassegna il valore della variabile
    print(f"Il fattoriale di {numero} è {fattoriale}")#l'f string serve per creare per l'appunto una stringa ma senza dover chiudere tutte le parentesi e le virgolette ma mettendo semplicemente delle parentesi graffe
```

Ciao, attraverso questo programma puoi calcolare il fattoriale di qualsiasi numero che desideri sapere

Quindi, inserisci il numero intero positivo che desideri calcolare: 12
Il fattoriale di 12 è 479001600

2 LA SOMMA DEI PRIMI N NUMERI PARI

Il programma chiede all'utente di inserire un numero intero positivo N. Successivamente, calcola la somma dei primi N numeri pari e stampa il risultato.

```
[2]: #Chiedere all'utente di inserire un numero intero positivo N
print("Ciao, attraverso questo programma puoi calcolare la somma dei numeri pari di qualsiasi numero che desideri sapere")
```

```

N=int(input("Quindi, inserisci il numero intero positivo che desideri calcolare:"))
#Inizializzare la somma a zero
somma=0
#Calcolare la somma dei primi N numero pari
for numero in range(2,2*N+1,2):#il primo valore è sempre 2, il secondo valore è
    #la moltiplicazione del valore di N per 2 e aggiungendo poi 1 al risultato, il
    #terzo valore è sempre 2.
    somma+=numero#il simbolo += svolge due compiti in uno: fa la somma e
    #riassegna il valore della variabile
#Stampare la somma
print(f"La somma dei primi {N} numeri pari è {somma}")

```

Ciao, attraverso questo programma puoi calcolare la somma dei numeri pari di qualsiasi numero che desideri sapere
 Quindi, inserisci il numero intero positivo che desideri calcolare: 12
 La somma dei primi 12 numeri pari è 156

3 LA LISTA DEI NUMERI PARI

Il programma chiede all'utente di inserire un numero intero positivo N. Successivamente, genera una lista contenente tutti i numeri pari compresi tra $2 \times N$, compreso. Infine, stampa questa lista dei numeri pari.

[3]:

```

#Chiedere all'utente di inserire un numero intero positivo N
print("Ciao, attraverso questo programma puoi creare una lista dei numeri pari
      fino a qualsiasi numero che desideri sapere")
N=int(input("Quindi, inserisci il numero intero positivo da cui desideri creare
      la lista: "))
lista=[] #l'append serve a fare una lista da poter richiamare
#Calcolare la somma dei primi N numeri pari
for numero in range(2,2*N+1,2):
    lista.append(numero)#così si crea la lista che varia dalla variabile numero
print(lista)

```

Ciao, attraverso questo programma puoi creare una lista dei numeri pari fino a qualsiasi numero che desideri sapere
 Quindi, inserisci il numero intero positivo da cui desideri creare la lista: 12
 [2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24]

4 IL TIRO DEL DADO VIRTUALE DA INDOVINARE

Il programma simula un gioco in cui l'utente tenta di indovinare il numero del dado, fornendo feedback sull'esito dell'indovinello.

[3]:

```

#Comprendere l'utilizzo della libreria random creando un gioco virtuale
→interattivo
import random
print("Ciao, attraverso questo programma puoi simulare il lancio di un dado, u
→provando anche ad indovinare il numero")
#Genera un numero casuale da 1 a 6 (simulando così il lancio di un dado)
numerodeldado=random.randint(1,6) #random.randint vuol dire che il numero da u
→generare è compreso da 1 a 6
#Chiedi all'utente di indovinare il numero
indovinalnumero=int(input("Quindi, prova ad indovinare il numero che è uscito u
→nel dado (da 1 a 6): "))
#verifica se l'utente ha indovinato correttamente
if indovinalnumero==numerodeldado:
    print(f"Complimenti, hai indovinato! Infatti il numero del dado era u
→{numerodeldado}")
elif indovinalnumero<1 or indovinalnumero>6:#or vuol dire oppure
    print("Errore! Hai scritto un numero pari a 0 o superiore a 6, riprova con u
→un numero compreso da 1 a 5 riavviando prima il programma")
else:
    print(f"Mi dispiace, hai perso! Il numero del dado era {numerodeldado}")

```

Ciao, attraverso questo programma puoi simulare il lancio di un dado, provando anche ad indovinare il numero

Quindi, prova ad indovinare il numero che è uscito nel dado (da 1 a 6): 12
 Errore! Hai scritto un numero pari a 0 o superiore a 6, riprova con un numero compreso da 1 a 5 riavviando prima il programma

5 IL CONTEGGIO DELLE VOCALI IN UNA FRASE E LA CONVERSIONE DELLA FRASE IN MAIUSCOLO

Il programma permette all'utente di inserire una frase o una parola e restituire il numero di vocali presenti all'interno di essa. Utilizzando il confronto non case-sensitive e considerando le vocali accentate, il programma è in grado di contare correttamente le vocali indipendentemente dalla loro forma.

[2] :

```

#Chiedere all'utente di inserire una frase o una parola
print("Ciao, questo programma permette di contare le vocali presenti in una u
→frase")
frase=input("Quindi, inserisci una frase o una parola: ").lower()#converte tutto u
→in minuscolo per semplificare il conteggio e non dover limitare l'utente a u
→scrivere una determinata cosa
conteggiodellevocali=0
vocali="àáâäëéêëîíîïöóôöüùûü"
#Inizializzare il contatore o iteratore
for carattere in frase:
    if carattere in vocali:

```

```

conteggiodellevocali+=1
print(f"Nella frase/parola che hai appena scritto ci sono {conteggiodellevocali} vocali")

```

Ciao, questo programma permette di contare le vocali presenti in una frase
 Quindi, inserisci una frase o una parola: CIÀo sono MeZZo ùBRiàcò e sCRIVo così
 Nella frase/parola che hai appena scritto ci sono 16 vocali

6 IL CALCOLO STATISTICO DELLA NATALITÀ

Il programma permette all'utente di stimare l'andamento demografico della popolazione nel corso degli anni, basandosi su tassi di natalità e mortalità forniti.

```

[8]: #Inizializzare la popolazione e gli anni
print("Ciao, questo programma permette di simulare un calcolo statistico della natalità dal passato fino ad oggi attraverso dei dati matematici da inserire")
popolazione=int(input("Scrivi il numero della popolazione totale nel tuo paese in questo momento: "))
anni=int(input("Inserisci l'anno corrente: "))
tassodinatalita=float(input("Inserisci il tasso di natalità del tuo paese in questo momento: "))#vuol dire che l'input è un float cioè un numero con la virgola (si usa il punto) e non intero come è per l'int
tassodimortalita=float(input("Inserisci il tasso di mortalità del tuo paese in questo momento: "))
#Inizializzare il contatore o iteratore
for anno in range(anni):
    nascite=(popolazione*tassodinatalita)/100
    decessi=(popolazione*tassodimortalita)/100
    popolazione+=(nascite-decessi)
    print(f"Nell'anno {anno+1} la popolazione è stata di {int(popolazione)} abitanti")#si usa l'anno +1 così parte dall'anno dopo quello inserito dall'utente invece di quello inserito veramente dall'utente
if nascite>decessi:
    print("Dai dati statistici raccolti si può dedurre che nel corso degli anni ci sia stato un innalzamento demografico")
elif nascite==decessi:
    print("Dai dati statistici raccolti si può dedurre che nel corso degli anni il numero dei decessi non è cambiato rispetto a quello delle nascite, quindi c'è stata una crescita zero")
else:
    print("Dai dati statistici raccolti si può dedurre che nel corso degli anni ci sia stato un calo demografico")
print("Simulazione calcolo statistico della natalità dal passato fino ad oggi completato correttamente")

```

Ciao, questo programma permette di simulare un calcolo statistico della natalità dal passato fino ad oggi attraverso dei dati matematici da inserire

Scrivi il numero della popolazione totale nel tuo paese in questo momento: 12
 Inserisci l'anno corrente: 7
 Inserisci il tasso di natalità del tuo paese in questo momento: 3.5
 Inserisci il tasso di mortalità del tuo paese in questo momento: 5.7
 Nell'anno 1 la popolazione è stata di 11 abitanti
 Nell'anno 2 la popolazione è stata di 11 abitanti
 Nell'anno 3 la popolazione è stata di 11 abitanti
 Nell'anno 4 la popolazione è stata di 10 abitanti
 Nell'anno 5 la popolazione è stata di 10 abitanti
 Nell'anno 6 la popolazione è stata di 10 abitanti
 Nell'anno 7 la popolazione è stata di 10 abitanti
 Dai dati statistici raccolti si può dedurre che nel corso degli anni ci sia stato un calo demografico
 Simulazione calcolo statistico della natalità dal passato fino ad oggi completato correttamente

7 IL CALCOLO DELLA DATA GIORNALIERA

Il programma mostra all'utente la data e l'ora correnti, formattate in un formato specifico, utilizzando la libreria datetimedi Python.

```
[2]: #Comprendere come scrivere la data usando la libreria automatica datetime
import datetime#libreria automatica per la data
print("Ciao, questo programma permette di sapere la data, l'ora, i minuti e i secondi del momento preciso che si sta vivendo")
print("Quindi:")
oggi=datetime.datetime.today()#serve per ottenere la data e l'ora correnti
formattazionedata=oggi.strftime("%d/%m/%Y")#formattare la data nel formato "giorno/mese/anno"
formattazionetempo=oggi.strftime("%H:%M:%S")#formattare l'orario nel formato "ora:minuti:secondi"
print(f"Oggi è il giorno: {formattazionedata} e sono le ore:{formattazionetempo}")


```

Ciao, questo programma permette di sapere la data, l'ora, i minuti e i secondi del momento preciso che si sta vivendo
 Quindi:
 Oggi è il giorno: 16/10/2023 e sono le ore: 21:18:33

8 IL CONVERTITORE DI UNITÀ DI MISURA UNIVERSALE

Il programma offre una soluzione rapida per convertire tra diverse unità di misura, fornendo il risultato della conversione in base all'input fornito dall'utente.

```
[6]: #Programma che permette di convertire diverse unità di misura con altre per sapere il risultato
print("Ciao, benvenuto nel convertitore di unità di misura universale!")
```

```

scelta=input("Quindi adesso scrivi quale unità di misura desideri convertire
→(puoi scegliere soltanto tra queste opzioni: metri/piedi/chilogrammi/libbre/
→centimetri/pollici): ").lower()#converte tutto in minuscolo per semplificare
→il conteggio e non dover limitare l'utente a scrivere una determinata cosa
if scelta=="metri":#per ogni specificazione si una condizione diversa
    valore=float(input("Inserisci il specifico valore in metri: "))#vuol dire
→che l'input è un float cioè un numero con la virgola (si usa il punto) e non
→intero come è per l'int
    risultato=valore*3.2884#risultato finale del calcolo
    print(f"{valore} metri corrispondono a {risultato} piedi")
elif scelta=="piedi":
    valore=float(input("Inserisci il specifico valore in piedi: "))
    risultato=valore*3.8084
    print(f"{valore} piedi corrispondono a {risultato} metri")
elif scelta=="chilogrammi":
    valore=float(input("Inserisci il specifico valore in chilogrammi: "))
    risultato=valore*2.02462
    print(f"{valore} chilogrammi corrispondono a {risultato} libbre")
elif scelta=="libbre":
    valore=float(input("Inserisci il specifico valore in libbre: "))
    risultato=valore*2.20462
    print(f"{valore} libbre corrispondono a {risultato} chilogrammi")
elif scelta=="centimetri":
    valore=float(input("Inserisci il specifico valore in centimetri: "))
    risultato=valore*0.39370079
    print(f"{valore} centimetri corrispondono a {risultato} pollici")
elif scelta=="pollici":
    valore=float(input("Inserisci il specifico valore in pollici: "))
    risultato=valore*2.54
    print(f"{valore} pollici corrispondono a {risultato} centimetri")
else:
    print("Scelta non riconosciuta. Scegli tra una delle seguenti unità di
→misura riavviando prima il programma: metri/piedi/chilogrammi/libbre/
→centimetri/pollici")

```

Ciao, benvenuto nel convertitore di unità di misura universale!
Quindi adesso scrivi quale unità di misura desideri convertire (puoi scegliere
soltanto tra queste opzioni: metri/piedi/chilogrammi/libbre/centimetri/pollici):
POLLICI
Inserisci il specifico valore in pollici: 17.3
17.3 pollici corrispondono a 43.942 centimetri

9 IL CONVERTITORE DA CELSIUS/FAHRENHEIT/KELVIN

Il programma offre una soluzione rapida per convertire tra Celsius, Fahrenheit e Kelvin, fornendo il risultato della conversione in base all'input fornito dall'utente.

```
[15]: print("Ciao, benvenuto nel convertitore di unità di misura da gradi Celsius/\n"
         "→Fahrenheit/Kelvin!")
sceltainiziale=input("Quindi adesso scrivi DA quale unità di misura desideri\n"
                     "→convertire? (puoi scegliere soltanto tra queste opzioni: Celsius/Fahrenheit/\n"
                     "→Kelvin): ").lower()
sceltafinale=input("Quindi adesso scrivi IN quale unità di misura desideri\n"
                   "→convertire? (puoi scegliere soltanto tra queste opzioni: Celsius/Fahrenheit/\n"
                   "→Kelvin): ").lower()
if sceltainiziale=="celsius":
    if sceltafinale=="fahrenheit":
        celsius=float(input("Inserisci il specifico valore in gradi Celsius: "))
        fahrenheit=(celsius*9/5)+32
        print(f"{celsius}°C corrispondono a {fahrenheit}°F")
    elif sceltafinale=="kelvin":
        celsius=float(input("Inserisci il specifico valore in gradi Celsius: "))
        kelvin=celsius+273.15
        print(f"{celsius}°C corrispondono a {kelvin}K")
    else:
        print("Scelta finale non riconosciuta. Scegli tra Celsius, Fahrenheit o\n"
              "→Kelvin.")
elif sceltainiziale=="fahrenheit":
    if sceltafinale=="celsius":
        fahrenheit=float(input("Inserisci il specifico valore in gradi\n"
                               "→Fahrenheit: "))
        celsius=(fahrenheit-32)*5/9
        print(f"{fahrenheit}°F corrispondono a {celsius}°C")
    elif sceltafinale=="kelvin":
        fahrenheit=float(input("Inserisci il specifico valore in gradi\n"
                               "→Fahrenheit: "))
        kelvin=(fahrenheit-32)*5/9+273.15
        print(f"{fahrenheit}°F corrispondono a {kelvin}K")
    else:
        print("Scelta finale non riconosciuta. Scegli tra Celsius, Fahrenheit o\n"
              "→Kelvin.")
elif sceltainiziale=="kelvin":
    if sceltafinale=="celsius":
        kelvin=float(input("Inserisci il specifico valore in Kelvin: "))
        celsius=kelvin-273.15
        print(f"{kelvin}K corrispondono a {celsius}°C")
    elif sceltafinale=="fahrenheit":
        kelvin=float(input("Inserisci il specifico valore in Kelvin: "))
        fahrenheit=(kelvin-273.15)*9/5+32
        print(f"{kelvin}K corrispondono a {fahrenheit}°F")
    else:
        print("Scelta finale non riconosciuta. Scegli tra Celsius, Fahrenheit o\n"
              "→Kelvin.")
```

```

else:
    print("Scelta iniziale non riconosciuta. Scegli tra una delle seguenti unità di misura riavviando prima il programma: Celsius, Fahrenheit o Kelvin.")

```

Ciao, benvenuto nel convertitore di unità di misura da gradi Celsius/Fahrenheit/Kelvin!

Quindi adesso scrivi DA quale unità di misura desideri convertire? (puoi scegliere soltanto tra queste opzioni: Celsius/Fahrenheit/Kelvin): CELSIUS
 Quindi adesso scrivi IN quale unità di misura desideri convertire? (puoi scegliere soltanto tra queste opzioni: Celsius/Fahrenheit/Kelvin): KELVIN
 Inserisci il specifico valore in gradi Celsius: 12
 12.0°C corrispondono a 285.15K

10 L'ALGORITMO PER LA SEQUENZA DI FIBONACCI

Il programma calcola e restituisce l'n-esimo numero di Fibonacci per un valore n specificato dall'utente.

```
[2]: print("Ciao, attraverso questo programma puoi calcolare l'n-esimo numero di Fibonacci di qualsiasi numero che desideri sapere")
#Chiedere all'utente di inserire un numero n
n=int(input("Inserisci il numero di cui desideri calcolare l'n-esimo numero di Fibonacci: "))
a=0
b=1
if n<=0:
    print("Impossibile eseguire il calcolo perchè il numero deve essere maggiore di 0. Riavvia il programma per inserire un nuovo numero")
elif n==1:
    risultato=0
else:
    #Inizializzare il contatore o iteratore
    for iterazione in range(n-1):
        c=a+b
        a=b
        b=c
    risultato=c
print(f'L'n-esimo numero di Fibonacci di {n} è {risultato}')
```

Ciao, attraverso questo programma puoi calcolare l'n-esimo numero di Fibonacci di qualsiasi numero che desideri sapere
 Inserisci il numero di cui desideri calcolare l'n-esimo numero di Fibonacci: 12
 L'n-esimo numero di Fibonacci di 12 è 144

11 LE FUNZIONI CUSTOM CON FIBONACCI

Questa funzione genera la sequenza di Fibonacci fino all'esimo termine specificato e restituisce la serie completa come elenco.

```
[1]: def fibonacci(n):#definizione di una funzione chiamata "fibonacci" con un  
    ↪parametro "n" (si può chiamare come dir si voglia)  
        seriedifibonacci=[0,1]#inizializzazione della serie di Fibonacci con i primi  
    ↪due valori (0 e 1)  
        while len(seriedifibonacci)<n:#utilizzo di un ciclo "while" per generare la  
    ↪serie fino a raggiungere la lunghezza 'n'  
            seriedifibonacci.  
    ↪append(seriedifibonacci[-1]+seriedifibonacci[-2])#calcolo del prossimo valore  
    ↪della serie e lo si aggiunge alla lista  
        return seriedifibonacci#la funzione restituisce la serie di Fibonacci  
    ↪generata
```

```
[7]: print("Ciao, attraverso questo programma puoi calcolare la sequenza di Fibonacci  
    ↪di qualsiasi numero che desideri sapere")  
n=int(input("Quindi, inserisci il numero da cui desideri calcolare la sequenza  
    ↪di Fibonacci: "))  
if n<=0:  
    n=input("Inserisci un numero positivo: ")  
else:  
    risultato=fibonacci(n)#calcola la sequenza di Fibonacci del numero n e  
    ↪assegna il risultato a "risultato"  
    print(f"Essendo che il numero è {n} allora la sequenza sarà: {risultato}")
```

Ciao, attraverso questo programma puoi calcolare la sequenza di Fibonacci di qualsiasi numero che desideri sapere
Quindi, inserisci il numero da cui desideri calcolare la sequenza di Fibonacci:
12
Essendo che il numero è 12 allora la sequenza sarà: [0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89]

12 IL CALCOLATORE DI AREE GEOMETRICHE

Questo programma fornisce una soluzione rapida per calcolare l'area di diverse forme geometriche, consentendo all'utente di effettuare la scelta e inserire le dimensioni necessarie.

```
[14]: import math  
def calcoloareadelcerchio(raggio):#definizione della funzione  
    return math.pi*(raggio**2)  
def calcoloareatriangolo(base, altezza):#definizione della funzione  
    return base*altezza/2  
def calcoloareadelrettangolo(base, altezza):#definizione della funzione  
    return base*altezza
```

```

def calcolaareaquadrato(lato):
    return lato*lato
print("Ciao, benvenuto nel calcolatore di aree geometriche!")
sceltaarea=input("Desideri calcolare l'area di un cerchio (c), del rettangolo
→(r), del triangolo (t) o del quadrato (q)? ").lower()
if sceltaarea=="c":
    raggio=float(input("Inserisci il raggio del cerchio: "))
    area=calcoloareadelcerchio(raggio)
    print(f"L'area del cerchio è {area:.2f} m^2")
elif sceltaarea=="r":
    base=float(input("Inserisci la base del rettangolo: "))
    altezza=float(input("Inserisci l'altezza del rettangolo: "))
    area=calculoareadelrettangolo(base, altezza)
    print(f"L'area del rettangolo è {area:.2f} m^2")
elif sceltaarea=="t":
    base=float(input("Inserisci la base del triangolo: "))
    altezza=float(input("Inserisci l'altezza del triangolo: "))
    area=calculoareatriangolo(base, altezza)
    print(f"L'area del triangolo è {area:.2f} m^2")
elif sceltaarea=="q":
    lato=float(input("Inserisci la misura del lato: "))
    area=calcolaareaquadrato(lato)
    print(f"L'area del quadrato è: {area:.2f} m^2")
else:
    print("Scelta non riconosciuta. Scegli tra una delle seguenti opzioni
→riavviando prima il programma: cerchio (c), del rettangolo (r), del triangolo
→(t) o del quadrato (q) ")

```

Ciao, benvenuto nel calcolatore di aree geometriche!
 Desideri calcolare l'area di un cerchio (c), del rettangolo (r), del triangolo
 (t) o del quadrato (q)? R
 Inserisci la base del rettangolo: 12
 Inserisci l'altezza del rettangolo: 7
 L'area del rettangolo è 84.00 m^2

13 IL CALCOLATORE DI INTERESSI

Questo programma fornisce una soluzione rapida per calcolare l'importo finale di un investimento utilizzando l'interesse composto, consentendo all'utente di inserire l'importo iniziale, il tasso di interesse annuale e il periodo di investimento.

```
[13]: def calcolainteresse(importoiniziale, tassointeresse, periodoinvestimento):
    importofinale=importoiniziale*(1+tassointeresse/100)**periodoinvestimento
    return importofinale
print("Ciao, benvenuto nel calcolatore di interessi!")
importoiniziale=float(input("Inserisci l'importo iniziale: "))
tassointeresse=float(input("Inserisci il tasso di interesse annuale (in %): "))
```

```

periodo_investimento=float(input("Inserisci il periodo di investimento (in anni):"))
importo_finale=calcola_interesse(importo_iniziale, tasso_interesse, periodo_investimento)
print(f'L'importo finale è {importo_finale}')

```

Ciao, benvenuto nel calcolatore di interessi!
 Inserisci l'importo iniziale: 3
 Inserisci il tasso di interesse annuale (in %): 12
 Inserisci il periodo di investimento (in anni): 55
 L'importo finale è 1527.9618170204692

14 IL CALCOLATORE DELLA FORZA GRAVITAZIONALE TRA PIANETI

Questo programma fornisce un modo per calcolare la forza gravitazionale tra la Terra e i pianeti selezionati dal sistema solare, consentendo all'utente di effettuare la scelta e ottenere il risultato desiderato.

```

[5]: def forza_gravitazionale(m1, m2, r):
    # Costante gravitazionale
    G=6.67e-11
    F=G*m1*m2/r**2
    return F
# Distanze tra la Terra e gli altri pianeti in metri
distanze={
    "Terra": 0,
    "Luna": 384400000,
    "Marte": 225000000000,
    "Giove": 778300000000
}
# Masse dei pianeti in kg
m1=5.97e24
m2={
    "Terra": 5.97e24,
    "Luna": 7.34e22,
    "Marte": 6.39e23,
    "Giove": 1.89e27
}
print("Ciao, benvenuto nel calcolatore della forza gravitazionale tra pianeti!")
while True:
    print("Pianeti disponibili:")
    print("1. Terra")
    print("2. Luna")
    print("3. Marte")
    print("4. Giove")
    print("5. Esci")

```

```

scelta=input("Quindi scegli un pianeta scrivendo il suo numero correlato (1/
→2/3/4/5): ")
if scelta=="1":
    pianeta="Terra"
elif scelta=="2":
    pianeta="Luna"
elif scelta=="3":
    pianeta="Marte"
elif scelta=="4":
    pianeta="Giove"
elif scelta=="5":
    print("Arrivederci e grazie per aver usato questo programma!")
    break
else:
    print("Scelta non valida. Riprova. Puoi scegliere solo una delle tre_
→opzioni (1, 2, 3, 4, 5)")
    continue
distanza=distanze.get(pianeta, None)
if distanza is not None:
    if distanza==0:
        print(f"La distanza tra Terra e {pianeta} è zero. La forza_
→gravitazionale è infinita.")
    else:
        forza=forza_gravitazionale(m1, m2[pianeta], distanza)
        print(f"La forza gravitazionale tra Terra e {pianeta} è: {forza}_
→Newton")

```

Ciao, benvenuto nel calcolatore della forza gravitazionale tra pianeti!

Pianeti disponibili:

1. Terra
2. Luna
3. Marte
4. Giove
5. Esci

Quindi scegli un pianeta scrivendo il suo numero correlato (1/2/3/4/5): 1

La distanza tra Terra e Terra è zero. La forza gravitazionale è infinita.

Pianeti disponibili:

1. Terra
2. Luna
3. Marte
4. Giove
5. Esci

Quindi scegli un pianeta scrivendo il suo numero correlato (1/2/3/4/5): 2

La forza gravitazionale tra Terra e Luna è: 1.978014482074582e+20 Newton

Pianeti disponibili:

1. Terra
2. Luna

3. Marte
 4. Giove
 5. Esci
 Quindi scegli un pianeta scrivendo il suo numero correlato (1/2/3/4/5): 3
 La forza gravitazionale tra Terra e Marte è: 5026156266666666.0 Newton
 Pianeti disponibili:
 1. Terra
 2. Luna
 3. Marte
 4. Giove
 5. Esci
 Quindi scegli un pianeta scrivendo il suo numero correlato (1/2/3/4/5): 4
 La forza gravitazionale tra Terra e Giove è: 1.242418496487888e+18 Newton
 Pianeti disponibili:
 1. Terra
 2. Luna
 3. Marte
 4. Giove
 5. Esci
 Quindi scegli un pianeta scrivendo il suo numero correlato (1/2/3/4/5): 5
 Arrivederci e grazie per aver usato questo programma!

15 IL RISOLTORE DI ANAGRAMMI

Questo programma fornisce una soluzione per trovare gli anagrammi di una parola o di una frase inserita dall'utente, stampando tutti gli anagrammi e il loro numero totale.

```
[41]: from itertools import permutations
k=0

def trovaanagrammi(parola):#definizione della funzione
    anagrammi=[''.join(p) for p in permutations(parola)]
    return anagrammi
print("Ciao, benvenuto nel risolutore di anagrammi!")
parolaofrase=input("Inserisci la parola o la frase di cui vuoi sapere\u
→l'anagramma: ").strip().lower()#strip vuol dire che il programma cancella gli\u
→spazi prima della prima parola e quelli dopo l'ultima parola se ci fossero
if len(parolaofrase)<2:
    print("Inserisci una parola o una frase con almeno 2 caratteri! Per\u
→riprovare riavvia il programma")
else:
    anagrammi=trovaanagrammi(parolaofrase)
    for anagramma in anagrammi:
        if anagramma!=parolaofrase:
            k+=1
            print(anagramma)
    print(f"Gli anagrammi di '{parolaofrase}' sono: '{k}'")
```

Ciao, benvenuto nel risolutore di anagrammi!

Inserisci la parola o la frase di cui vuoi sapere l'anagramma: Matteo

mattoe

mateto

mateot

matote

matoet

mattoe

mateto

mateot

matote

matoet

maetto

maetot

maetto

maetot

maeott

maeott

maotte

maotet

maotte

maotet

maoett

maoett

mtateo

mtatoe

mtaeto

mtaeot

mtaote

mtaoet

mttaeo

mttaoe

mtteao

mtteoa

mttoae

mttoea

mteato

mteaot

mtetao

mtetoa

mteoat

mteota

mtoate

mtoaet

mtotae

mtotea

mtoeat

mtoeta

mtateo
mtatoe
mtaeto
mtaeot
mtaote
mtaoet
mttaeo
mttaoe
mtteao
mtteoa
mttoae
mttoea
mteato
mteaot
mtetao
mtetoa
mteoat
mteota
mtoate
mtoaet
mtotae
mtotea
mtoeat
mtoeta
meatto
meatot
meatot
meaott
meaott
metato
metaot
mettao
mettoa
metoat
metota
metato
metaot
mettao
mettoa
metoat
metota
meoatt
meoatt
meotat
meotta
meotat
meotta

moatte
moatet
moatte
moatet
moaett
moaett
motate
motaet
mottae
mottea
moteat
moteta
motate
motaet
mottae
mottea
moteat
moteta
moeatt
moeatt
moetat
moetta
moetat
moetta
amtteo
amttoe
amteto
amteot
amtote
amtoet
amtteo
amttoo
amteto
amteot
amtote
amtoet
ametto
ametot
ameott
ameott
amotte
amotet
amotte
amotet
amoett
amoett

atmteo
atmtoe
atmeto
atmeot
atmote
atmoet
attmeo
attmoe
attemo
atteom
attome
attoem
atemto
atemot
atetmo
atetom
ateomt
ateotm
atomte
atomet
atotme
atotem
atoemt
atoetm
atmteo
atmtoe
atmeto
atmeot
atmote
atmoet
attmeo
attmoe
attemo
atteom
attome
attoem
atemto
atemot
atetmo
atetom
ateomt
ateotm
atomte
atomet
atotme
atotem
atoemt
atoetm

aemtto
aemtot
aemtto
aemtot
aemott
aemott
aetmto
aetmot
aettmo
aettom
aetomt
aetotm
aetmto
aetmot
aettmo
aettom
aetomt
aetotm
aeomtt
aeomtt
aeotmt
aeottm
aeotmt
aeottm
aomtte
aomtet
aomttes
aomttes
aomett
aotmte
aotmet
aottme
aottem
aotemt
aotetm
aotmte
aotmet
aottme
aottem
aotemt
aotetm
aoemtt
aoemtt
aoetmt
aoettm
aoetmt
aoettm

tmateo
tmatoe
tmaeto
tmaeot
tmaote
tmaoet
tmtaeo
tmtaoe
tmteao
tmteoa
tmtoae
tmtoea
tmeato
tmeaot
tmetao
tmetoa
tmeoat
tmeota
tmoate
tmoaet
tmotae
tmotea
tmoeat
tmoeta
tamteo
tamtoe
tameto
tameot
tamote
tamoet
tatmeo
tatmoe
tatemo
tateom
tatome
tatoem
taemto
taemot
taetmo
taetom
taeomt
taeotm
taomte
taomet
taotme
taotem
taoemt
taoetm

ttmaeo
ttmaoe
ttmeao
ttmeoa
ttmoae
ttmoea
ttameo
ttamoe
ttaemo
ttaeom
ttaome
ttaoem
ttemao
tteamo
tteaom
tteoma
tdeoam
ttomae
ttomea
ttoame
ttoaem
ttoema
ttoeam
temato
temaot
temtao
temtoa
temoat
temota
teamto
teamot
teatmo
teatom
teaomt
teaotm
tetmao
tetmoa
tetamo
tetaom
tetoma
tetoam
teomat
teomta
teoamt
teoatm
teotma
teotam

tomate
tomaet
tomtae
tomtea
tomeat
tometa
toamte
toamet
toatme
toatem
toaemt
toaetm
totmae
totmea
totame
totaem
totema
toteam
toemat
toemta
toeamt
toeatm
toetma
toetam
tmateo
tmatoe
tmaeto
tmaeot
tmaote
tmaoet
tmtaeo
tmtaoe
tmteao
tmteoa
tmtoae
tmtoea
tmeato
tmeaot
tmetao
tmetoa
tmeoat
tmeota
tmoate
tmoaet
tmotae
tmotea
tmoeat
tmoeta

tamteo
tamtoe
tameto
tameot
tamote
tamoet
tatmeo
tatmoe
tatemo
tateom
tatome
tatoem
taemto
taemot
taetmo
taetom
taeomt
taeotm
taomte
taomet
taotme
taotem
taoemt
taoetm
ttmaeo
ttmaoe
ttmeao
ttmoae
ttmoea
ttameo
ttamoe
ttaemo
ttaeom
ttaome
ttaoem
ttemao
ttemoa
tteamo
tteaom
tteoma
tteeam
ttomae
ttomea
ttoame
ttoaem
ttoema
ttoeam

temato
temaot
temtao
temtoa
temoat
temota
teamto
teamot
teatmo
teatom
teaomt
teaotm
tetmao
tetmoa
tetamo
tetaom
tetoma
tetoam
teomat
teomta
teoamt
teoatm
teotma
teotam
tomate
tomaet
tomtae
tomtea
tomeat
tometa
toamte
toamet
toatme
toatem
toaemt
toaetm
totmae
totmea
totame
totaem
totema
toteam
toemat
toemta
toeamt
toeatm
toetma
toetam

ematto
ematot
ematto
ematot
emaott
emaott
emtato
emtaot
emttao
emttoa
emtoat
emtota
emtato
emtaot
emttao
emttoa
emtoat
emtota
emoatt
emoatt
emotat
emotta
emotat
emotta
eamtto
eamtot
eamtto
eamtot
eamott
eatmto
eatmot
eattmo
eattom
eatomt
eatotm
eatmto
eatmot
eattmo
eattom
eatomt
eatotm
eaomtt
eaomtt
eaotmt
eaottm
eaotmt
eaottm

etmato
etmaot
etmtao
etmtoa
etmoat
etmota
etamto
etamot
etatmo
etatom
etaomt
etaotm
ettmao
ettmoa
ettamo
ettaom
ettoma
ettoam
etomat
etomta
etoamt
etoatm
etotma
etotam
etmato
etmaot
etmtao
etmtoa
etmoat
etmota
etamto
etamot
etatmo
etatom
etaomt
etaotm
ettmao
ettmoa
ettamo
ettaom
ettoma
ettoam
etomat
etomta
etoamt
etoatm
etotma
etotam

eomatt
eomatt
eomtat
eomtta
eomtat
eomtta
eoamtt
eoamtt
eoatmt
eoattm
eoatmt
eoattm
eotmat
eotmta
eotamt
eotatm
eottma
eottam
eotmat
eotmta
eotamt
eotatm
eottma
eottam
omatte
omatet
omatte
omatet
omaett
omaett
omtate
omtaet
omttae
omttea
omteat
omteta
omtate
omtaet
omttae
omttea
omteat
omteta
omeatt
omeatt
ometat
ometta
ometat
ometta

oamtte
oamtet
oamtte
oamtet
oamett
oamett
oatmte
oatmet
oattme
oattem
oatemt
oatetm
oatmte
oatmet
oattme
oattem
oatemt
oatetm
oaemtt
oaemtt
oaetmt
oaettm
oaetmt
oaettm
otmate
otmaet
otmtae
otmtea
otmeat
otmeta
otamte
otamet
otatme
otatem
otaemt
otaetm
ottmae
ottmea
ottame
ottaem
ottema
otteam
otemat
otemta
oteamt
oteatm
otetma
otetam

otmate
otmaet
otmtae
otmtea
otmeat
otmeta
otamte
otamet
otatme
otatem
otaemt
otaetm
ottmae
ottmea
ottame
ottaem
ottema
otteam
otemat
otemta
oteamt
oteatm
otetma
otetam
oematt
oematt
oemtat
oemtta
oemtat
oemtta
oeamtt
oeamtt
oeatmt
oeattm
oeatmt
oeattm
oetmat
oetmta
oetamt
oetatm
oettma
oettam
oetmat
oetmta
oetamt
oetatm
oettma
oettam

Gli anagrammi di 'matteo' sono: '718'

16 IL RISOLUTORE DI EQUAZIONI DI PRIMO GRADO (CON LA FUNZIONE MAIN)

Questo programma offre un'interfaccia interattiva per risolvere equazioni di primo grado e guidare l'utente attraverso il processo di risoluzione, fornendo la soluzione e offrendo la possibilità di risolvere ulteriori equazioni.

```
[30]: #Programma che permette di risolvere le equazioni di primo grado inserendo il
      ↵valore dei coefficienti
def risolutoreequazionidiprimogrado():#definizione della funzione
    while True:
        print("Ciao, questo programma permette di risolvere equazioni di primo
              ↵grado nella seguente formula: ax + b = 0")
        #Chiedere l'input dei coefficienti a e b
        a=float(input("Inserisci il coefficiente a: "))
        b=float(input("Inserisci il coefficiente b: "))
        #Stampare l'equazione di partenza
        equazione=input(f"Quindi l'equazione data è {a}x + {b} = 0 ? (rispondere
              ↵solo con sì o no): ").lower()#converte tutto in minuscolo per semplificare il
              ↵conteggio e non dover limitare l'utente a scrivere una determinata cosa
        if equazione=="si":
            print("Ok, allora il programma può procedere con la risoluzione
                  ↵dell'equazione")
        elif equazione=="no":
            print("Ok, allora adesso ripartirà automaticamente il programma da
                  ↵capo")
            continue#fa riavviare il programma da capo
        else:
            print("Risposta non riconosciuta. Ti ricordo che puoi solo
                  ↵rispondere con sì o no")
            print("Ok, allora adesso non avendo ricevuto una risposta tra quelle
                  ↵indicate precedentemente il programma ripartirà automaticamente da capo")
            continue
        #Verificare se l'equazione è lineare (cioè diversa da zero)
        if a==0 and b==0:#L'and serve a comparare più cose contemporaneamente
            print("L'equazione è indeterminata, infatti: 0x = 0")
        elif a==0:
            print(f"L'equazione è impossibile, infatti: 0x = {b}")
        else:
            #Calcolare la soluzione di x
            print("Passaggi per risolvere l'equazione: ")
            print(f"1) Per prima cosa sottrai {b} da entrambi i lati
                  ↵dell'equazione")
            print(f"2) Poi bisogna fare {a}x + {b} - {b} = 0 - {b}")


```

```

#Dividere entrambi i lati per a
x=-b/a
print(f"4) Adesso bisogna dividere entrambi i lati per {a}")
print(f"5) Poi bisogna fare x = {-b}/{a}")
#Calcolare il valore di x
print(f"6) Adesso bisogna calcolare il valore di x")
print(f"La soluzione della equazione quindi è x>{x}")
scelta=input("Si desidera risolvere un'altra equazione? (rispondere solo con si o no): ").lower()
if scelta=="si":
    print("Il programma si riavvierà tra poco")
    continue
elif scelta=="no":
    print("Grazie per aver usato questo programma")
    break#fa fermare il programma
else:
    print("Risposta non riconosciuta. Ti ricordo che puoi solo rispondere con si e no")
    print("Grazie per aver usato questo programma")
    print("Ok, allora adesso non avendo ricevuto una risposta tra quelle indicate precedentemente il programma ripartirà automaticamente da capo")
if __name__ == "__main__":# Questa è una condizione che verifica se il file Python è in esecuzione come script principale
    risolutorreequazionidiprimogrado()
    #Il programma chiama la funzione "risolutorreequazionidiprimogrado()" solo se il file è eseguito come script principale.

```

Ciao, questo programma permette di risolvere equazioni di primo grado nella seguente formula: $ax + b = 0$

Inserisci il coefficiente a : 3

Inserisci il coefficiente b : 12

Quindi l'equazione data è $3.0x + 12.0 = 0$? (rispondere solo con si o no): SI

Ok, allora il programma può procedere con la risoluzione dell'equazione

Passaggi per risolvere l'equazione:

- 1) Per prima cosa sottrai 12.0 da entrambi i lati dell'equazione
- 2) Poi bisogna fare $3.0x + 12.0 - 12.0 = 0 - 12.0$
- 4) Adesso bisogna dividere entrambi i lati per 3.0
- 5) Poi bisogna fare $x = -12.0/3.0$
- 6) Adesso bisogna calcolare il valore di x

La soluzione della equazione quindi è $x > -4.0$

Si desidera risolvere un'altra equazione? (rispondere solo con si o no): NON

Risposta non riconosciuta. Ti ricordo che puoi solo rispondere con si e no

Grazie per aver usato questo programma

Ok, allora adesso non avendo ricevuto una risposta tra quelle indicate precedentemente il programma ripartirà automaticamente da capo

Ciao, questo programma permette di risolvere equazioni di primo grado nella seguente formula: $ax + b = 0$

Inserisci il coefficiente a: 7
 Inserisci il coefficiente b: 12
 Quindi l'equazione data è $7.0x + 12.0 = 0$? (rispondere solo con si o no): SI
 Ok, allora il programma può procedere con la risoluzione dell'equazione
 Passaggi per risolvere l'equazione:
 1) Per prima cosa sottrai 12.0 da entrambi i lati dell'equazione
 2) Poi bisogna fare $7.0x + 12.0 - 12.0 = 0 - 12.0$
 4) Adesso bisogna dividere entrambi i lati per 7.0
 5) Poi bisogna fare $x = -12.0/7.0$
 6) Adesso bisogna calcolare il valore di x
 La soluzione della equazione quindi è $x > -1.7142857142857142$
 Si desidera risolvere un'altra equazione? (rispondere solo con si o no): NO
 Grazie per aver usato questo programma

17 IL RISOLUTORE DI DISEQUAZIONI DI PRIMO GRADO (CON LA FUNZIONE MAIN)

Questo programma offre un'interfaccia interattiva per risolvere disequazioni di primo grado e guidare l'utente attraverso il processo di risoluzione, fornendo la soluzione e offrendo la possibilità di risolvere ulteriori disequazioni.

```
[6]: def risolutoredisequazionidiprimogrado():
    while True:
        print("Ciao! Questo programma permette di risolvere disequazioni di primo grado nella seguente formula: ax + b>0")
        a=float(input("Inserisci il coefficiente a: "))
        b=float(input("Inserisci il coefficiente b: "))
        disequazione=input(f"Quindi la disequazione data è {a}x + {b}>0? (rispondi solo con si o no): ").lower()
        if disequazione=="si":
            print("Ok, il programma può procedere con la risoluzione della disequazione")
        elif disequazione=="no":
            print("Ok, il programma si riavvierà automaticamente")
            continue
        else:
            print("Risposta non riconosciuta. Puoi rispondere solo con si o no.")
            print("Il programma si riavvierà automaticamente.")
            continue
        #Verifica se la disequazione è valida
        if a==0:
            print("La disequazione non è di primo grado.")
        else:
            #Calcola la soluzione di x
            x=-b/a
            #Spiegazione dettagliata dei passaggi matematici
            print("Passaggi per risolvere la disequazione:")

```

```

        print(f"1) Dividi entrambi i lati per {a}, ricordando di invertire il segno della disequazione, quindi la disequazione diventa:")
        print(f"    {a}x + {b}>0 diventa x>{x}")
        print(f"La soluzione della disequazione quindi è x>{x}")
        scelta=input("Vuoi risolvere un'altra disequazione? (rispondi solo con si o no): ").lower()
        if scelta=="si":
            print("Il programma si riavvierà tra poco.")
            continue
        elif scelta=="no":
            print("Grazie per aver usato questo programma.")
            break
        else:
            print("Risposta non riconosciuta. Puoi rispondere solo con si o no.")
            print("Grazie per aver usato questo programma.")
            print("Il programma si riavvierà automaticamente.")
if __name__ == "__main__":
    risolutoredisequazionidiprimogrado()

```

Ciao! Questo programma permette di risolvere disequazioni di primo grado nella seguente formula: $ax + b > 0$

Inserisci il coefficiente a: 3

Inserisci il coefficiente b: 12

Quindi la disequazione data è $3.0x + 12.0 > 0$? (rispondi solo con si o no): NO

Ok, il programma si riavvierà automaticamente

Ciao! Questo programma permette di risolvere disequazioni di primo grado nella seguente formula: $ax + b > 0$

Inserisci il coefficiente a: -0.5

Inserisci il coefficiente b: 2

Quindi la disequazione data è $-0.5x + 2.0 > 0$? (rispondi solo con si o no): SI

Ok, il programma può procedere con la risoluzione della disequazione

Passaggi per risolvere la disequazione:

1) Dividi entrambi i lati per -0.5, ricordando di invertire il segno della disequazione, quindi la disequazione diventa:

$-0.5x + 2.0 > 0$ diventa $x > 4.0$

La soluzione della disequazione quindi è $x > 4.0$

Vuoi risolvere un'altra disequazione? (rispondi solo con si o no): SI

Il programma si riavvierà tra poco.

Ciao! Questo programma permette di risolvere disequazioni di primo grado nella seguente formula: $ax + b > 0$

Inserisci il coefficiente a: 1.5

Inserisci il coefficiente b: 1

Quindi la disequazione data è $1.5x + 1.0 > 0$? (rispondi solo con si o no): NON

Risposta non riconosciuta. Puoi rispondere solo con sì o no.

Il programma si riavvierà automaticamente.

Ciao! Questo programma permette di risolvere disequazioni di primo grado nella seguente formula: $ax + b > 0$

Inserisci il coefficiente a: 1
 Inserisci il coefficiente b: 1.5
 Quindi la disequazione data è $1.0x + 1.5 > 0$? (rispondi solo con si o no): SI
 Ok, il programma può procedere con la risoluzione della disequazione
 Passaggi per risolvere la disequazione:
 1) Dividi entrambi i lati per 1.0, ricordando di invertire il segno della disequazione, quindi la disequazione diventa:
 $1.0x + 1.5 > 0$ diventa $x > -1.5$
 La soluzione della disequazione quindi è $x > -1.5$
 Vuoi risolvere un'altra disequazione? (rispondi solo con si o no): NO
 Grazie per aver usato questo programma.

18 IL RISOLUTORE DI EQUAZIONI LINEARI DI PRIMO GRADO (CON LA FUNZIONE MAIN)

Questo programma fornisce un'interfaccia interattiva per risolvere equazioni lineari e guida l'utente attraverso il processo di risoluzione, fornendo la soluzione e offrendo la possibilità di risolvere ulteriori equazioni.

```
[7]: def risolutoreequazionelineare():
    while True:
        print("Ciao! Questo programma permette di risolvere equazioni lineari nella seguente formula: ax + b = 0")
        a=float(input("Inserisci il coefficiente a: "))
        b=float(input("Inserisci il coefficiente b: "))
        equazione=input(f"Quindi l'equazione data è {a}x + {b} = 0? (rispondi solo con si o no): ").lower()
        if equazione=="si":
            print("Ok, il programma può procedere con la risoluzione dell'equazione")
        elif equazione=="no":
            print("Ok, il programma si riavvierà automaticamente")
            continue
        else:
            print("Risposta non riconosciuta. Puoi rispondere solo con sì o no.")
            print("Il programma si riavvierà automaticamente.")
            continue
        # Verifica se l'equazione è lineare (cioè diversa da zero)
        if a==0 and b==0:
            print("L'equazione è indeterminata, infatti: 0x = 0")
        elif a==0:
            print(f"L'equazione è impossibile, infatti: 0x = {b}")
        else:
            # Calcola la soluzione di x
            x=-b/a
            # Aggiunta della spiegazione dettagliata dei passaggi matematici
            print("Passaggi per risolvere l'equazione:")

```

```

        print(f"1) Dividi entrambi i lati per {a}, quindi l'equazione diventa:")
        print(f"    {a}x + {b} = 0 diventa x = {-b}/{a}")
        print(f"La soluzione dell'equazione quindi è x = {x}")
    scelta=input("Vuoi risolvere un'altra equazione lineare? (rispondi solo con si o no): ").lower()
    if scelta=="si":
        print("Il programma si riavvierà tra poco.")
        continue
    elif scelta=="no":
        print("Grazie per aver usato questo programma.")
        break
    else:
        print("Risposta non riconosciuta. Puoi rispondere solo con si o no.")
        print("Grazie per aver usato questo programma.")
        print("Il programma si riavvierà automaticamente.")
if __name__ == "__main__":
    risolutoreequazionelineare()

```

Ciao! Questo programma permette di risolvere equazioni lineari nella seguente formula: $ax + b = 0$

Inserisci il coefficiente a: 2

Inserisci il coefficiente b: -6

Quindi l'equazione data è $2.0x + -6.0 = 0$? (rispondi solo con si o no): SI

Ok, il programma può procedere con la risoluzione dell'equazione

Passaggi per risolvere l'equazione:

- 1) Dividi entrambi i lati per 2.0, quindi l'equazione diventa:
 $2.0x + -6.0 = 0$ diventa $x = 6.0/2.0$

La soluzione dell'equazione quindi è $x = 3.0$

Vuoi risolvere un'altra equazione lineare? (rispondi solo con si o no): SI

Il programma si riavvierà tra poco.

Ciao! Questo programma permette di risolvere equazioni lineari nella seguente formula: $ax + b = 0$

Inserisci il coefficiente a: 0.5

Inserisci il coefficiente b: 1

Quindi l'equazione data è $0.5x + 1.0 = 0$? (rispondi solo con si o no): NO

Ok, il programma si riavvierà automaticamente

Ciao! Questo programma permette di risolvere equazioni lineari nella seguente formula: $ax + b = 0$

Inserisci il coefficiente a: 0.5

Inserisci il coefficiente b: 1

Quindi l'equazione data è $0.5x + 1.0 = 0$? (rispondi solo con si o no): SI

Ok, il programma può procedere con la risoluzione dell'equazione

Passaggi per risolvere l'equazione:

- 1) Dividi entrambi i lati per 0.5, quindi l'equazione diventa:
 $0.5x + 1.0 = 0$ diventa $x = -1.0/0.5$

La soluzione dell'equazione quindi è $x = -2.0$

Vuoi risolvere un'altra equazione lineare? (rispondi solo con si o no): NO
Grazie per aver usato questo programma.

19 I DIZIONARI E IL RISOLUTORE DI DEFINIZIONI DEI TASSI DI CAMBIO

Questo programma fornisce un modo semplice per convertire importi tra diverse valute utilizzando i tassi di cambio giornalieri.

```
[16]: #Definizione dei tassi di cambio
tassidicambio={#definizione del dizionario
    "dollari": 1.0,#key
    "euro": 0.85,#key
    "yen": 110.41,#key
    "lire italiane": 1936.27,#key
    "franco svizzero": 0.96#key
    #Aggiungere altre valute e tassi di cambio se è necessario
}
print("Ciao, con questo programma puoi vedere il tasso giornaliero delle
    ↪seguenti valute: dollari, euro, yen, lire italiane e il franco svizzero")
#Chiedere all'utente l'importo delle varie valute per capire il cambio
importo=float(input("Inserisci l'importo che desideri convertire: "))
valutadipartenza=input("Inserisci la valuta di partenza (puoi scegliere solo:
    ↪dollari, euro, yen, lire italiane e il franco svizzero): ").lower()#converte
    ↪tutto in minuscolo per semplificare il conteggio e non dover limitare l'utente
    ↪a scrivere una determinata cosa
valutadidestinazione=input("Inserisci la valuta di destinazione (puoi scegliere
    ↪solo: dollari, euro, yen, lire italiane e il franco svizzero): ").lower()
if valutadipartenza in tassidicambio and valutadidestinazione in tassidicambio:
    tassodicambio=tassidicambio[valutadidestinazione]/
    ↪tassidicambio[valutadipartenza]#Le parentesi quadre [ ] vengono utilizzate per
    ↪accedere (richiamare) a un valore in un dizionario usando una keys (chiave),
    ↪nonchè "valori" corrispondenti all'interno del dizionario
    importoconvertito=importo*tassodicambio
    print(f"\{importo} {valutadipartenza} valgono e sono equivalenti a
        ↪\{importoconvertito:.2f} {valutadidestinazione}")# il simbolo 2f formatta
        ↪"importoconvertito" come un numero decimale a virgola mobile con due cifre
        ↪decimali (due perchè è 2f)
else:
    print("Valuta non supportata. Puoi inserire un'altra valuta riavviando prima
        ↪il programma (puoi scegliere solo: dollari, euro, yen, lire italiane e il
        ↪franco svizzero)")
```

Ciao, con questo programma puoi vedere il tasso giornaliero delle seguenti valute: dollari, euro, yen, lire italiane e il franco svizzero
Inserisci l'importo che desideri convertire: 2000
Inserisci la valuta di partenza (puoi scegliere solo: dollari, euro, yen, lire

italiane e il franco svizzero): LIRE ITALIANE
 Inserisci la valuta di destinazione (puoi scegliere solo: dollari, euro, yen,
 lire italiane e il franco svizzero): EURO
 2000.0 lire italiane valgono e sono equivalenti a 0.88 euro

20 IL RISOLUTORE DI DEFINIZIONI DEI TASSI DI CAMBIO (TRAMITE UNA LIBRERIA)

Il programma utilizza la libreria forex-python per ottenere i tassi di cambio giornalieri delle valute rispetto a una valuta di base specificata dall'utente

```
[8]: #!pip install forex-python (installare la libreria una sola volta per il funzionamento del programma togliendo l'asterisco e eliminando il commento)
from forex_python.converter import CurrencyRates#libreria necessaria per il programma
print("Ciao, con questo programma puoi vedere il tasso giornaliero di molte valute, come l'USD, l'EUR, il GBP, ecc...")
def ottienitassidicambio(valutadibase):#definizione tassi di cambio
    convertitore=CurrencyRates()#comandi della libreria
    tassidicambio=convertitore.get_rates(valutadibase)#comandi della libreria
    return tassidicambio#"return" restituisce il valore dell'espressione specificata quando la funzione viene chiamata
if __name__ == "__main__":
    valutadibase=input("Inserisci la valuta di base (es. USD, EUR, GBP): ").upper()#converte tutto in maiuscolo per semplificare il conteggio e non dover limitare l'utente a scrivere una determinata cosa
    #try e "except" consentono esclusivamente di gestire eccezioni e errori durante l'esecuzione del codice in modo controllato
    try:
        tassidicambio=ottienitassidicambio(valutadibase)
        print(f"*** Tassi di cambio rispetto a {valutadibase} ***")
        for valuta, tasso in tassidicambio.items():#"items()" permette di ottenere sia le chiavi che i valori di un dizionario all'interno di un ciclo (coppia chiave valore)
            print(f"1 {valutadibase} = {tasso} {valuta}")
    except:
        print("Valuta non valida o errore durante il recupero dei tassi di cambio. Riprova riavviando il programma e seguendo correttamente tutti i passaggi indicati a schermo")
```

Ciao, con questo programma puoi vedere il tasso giornaliero di molte valute, come l'USD, l'EUR, il GBP, ecc...

Inserisci la valuta di base (es. USD, EUR, GBP): EUR

*** Tassi di cambio rispetto a EUR ***

1 EUR = 1.0968 USD

1 EUR = 157.35 JPY

1 EUR = 1.9558 BGN

```

1 EUR = 24.293 CZK
1 EUR = 7.4511 DKK
1 EUR = 0.8618 GBP
1 EUR = 388.44 HUF
1 EUR = 4.467 PLN
1 EUR = 4.948 RON
1 EUR = 11.7195 SEK
1 EUR = 0.9627 CHF
1 EUR = 144.5 ISK
1 EUR = 11.2063 NOK
1 EUR = 29.6458 TRY
1 EUR = 1.6779 AUD
1 EUR = 5.3664 BRL
1 EUR = 1.474 CAD
1 EUR = 7.903 CNY
1 EUR = 8.5766 HKD
1 EUR = 16661.91 IDR
1 EUR = 90.8795 INR
1 EUR = 1441.47 KRW
1 EUR = 18.8106 MXN
1 EUR = 5.0146 MYR
1 EUR = 1.8103 NZD
1 EUR = 61.728 PHP
1 EUR = 1.4761 SGD
1 EUR = 38.361 THB
1 EUR = 20.8921 ZAR

```

21 IL CONTALETTERE

Questo programma conta quante volte ogni lettera dell'alfabeto appare nella frase inserita dall'utente e stampa il conteggio per ciascuna lettera, insieme al numero totale di lettere presenti nella frase.

```
[10]: #Programma che permette di contare quante lettere ci sono in una frase
print("Ciao, questo programma permettere di contare quante lettere ci sono in\u
      \u2192una frase")
#Chiedere all'utente di inserire una frase
frase=input("Quindi per prima cosa inserisci la frase che desideri che sia\u
            \u2192analizzata: ").lower()#converte tutto in minuscolo per semplificare il\u
            \u2192conteggio e non dover limitare l'utente a scrivere una determinata cosa
#Inizializzare una lista di lettere dell'alfabeto
alfabeto="abcdefghijklmnopqrstuvwxyz"
#Inizializzare un dizionario per tenere traccia del conteggio delle lettere
conteggiodellelettere={}
#Iterare attraverso ciascuna lettera dell'alfabeto
for lettera in alfabeto:
    #Contare quante volte appare la lettera nella frase

```

```

conteggio=frase.count(lettera) #conta quante volte la "lettera" appare nella frase".
#Aggiungere la lettera e il conteggio al dizionario se la lettera appare almeno una volta
if conteggio>0:
    conteggiodellelettere[lettera]=conteggio
#Stampare i conteggi delle singole lettere
for lettera, conteggio in conteggiodellelettere.items():
    print(f"{lettera}:{conteggio}")
#Calcolare il numero totale di lettere
numerototalelettere=sum(conteggiodellelettere.values())#serve a calcolare la somma dei valori all'interno del dizionario "conteggiodellelettere"
#"values()" restituisce una vista degli valori all'interno del dizionario 'conteggiodellelettere'.
#Stampare il conteggio totale
print(f"Il numero totale di lettere nella frase è {numerototalelettere}")

```

Ciao, questo programma permettere di contare quante lettere ci sono in una frase
Quindi per prima cosa inserisci la frase che desideri che sia analizzata: CIAO
SoNo matTeo

a:2
c:1
e:1
i:1
m:1
n:1
o:4
s:1
t:2

Il numero totale di lettere nella frase è 14

[12]: #Comprendere il comando items
conteggiodellelettere.items

[12]: <function dict.items>

[13]: #Comprendere il comando values
conteggiodellelettere.values

[13]: <function dict.values>

22 I FUSI ORARI

Questo programma, chiamato “Orologio Mondiale”, consente agli utenti di visualizzare l'orario attuale in diverse città del mondo. Ecco come funziona:

```
[2]: from datetime import datetime
import pytz
print("Ciao, benvenuto nell'orologio mondiale, con questo programma puoi vedere l'orario attuale di molte città del mondo!")
cittadisponibili={
    "New York": "America/New_York",
    "Londra": "Europe/London",
    "Tokyo": "Asia/Tokyo",
    "Sydney": "Australia/Sydney",
    "Rio de Janeiro": "America/Sao_Paulo",
    "Mosca": "Europe/Moscow",
    "Pechino": "Asia/Shanghai",
    "Delhi": "Asia/Kolkata",
    "Berlino": "Europe/Berlin",
    "Città del Messico": "America/Mexico_City",
    "Johannesburg": "Africa/Johannesburg",
    "Dubai": "Asia/Dubai",
    "Singapore": "Asia/Singapore"
}
while True:
    print("\nCittà disponibili: ")#con il simbolo /n si va a capo dentro ad una stringa di testo
    for citta in cittadisponibili.keys():
        print(citta)
    sceltadellacitta=input("\nInserisci il nome della città che desideri per visualizzare l'orario attuale (oppure scrivi 'esci' se desideri uscire dal programma): ").strip().title()#strip vuol dire che il programma cancella gli spazi prima della prima parola e quelli dopo l'ultima parola se ci fossero mentre title normalizza il testo rendendo corrette le maiuscole e le minuscole nelle parole
    if sceltadellacitta.lower()=="esci":
        print("Arrivederci e grazie per aver usato questo programma!")
        break
    elif sceltadellacitta in cittadisponibili:
        fusoorario=pytz.timezone(cittadisponibili[sceltadellacitta])
        orariocorrente=datetime.now(fusoorario)
        print(f'L'ora attuale a {sceltadellacitta} è {orariocorrente.strftime("%H:%M:%S")}\n')
    elif sceltadellacitta not in cittadisponibili:
        print("La città inserita non è nella lista. Per favore, prova di nuovo e controlla di aver scritto bene il nome della città (il nome deve essere scritto in maniera corretta per poter funzionare il programma).")
```

Ciao, benvenuto nell'orologio mondiale, con questo programma puoi vedere l'orario attuale di molte città del mondo!

Città disponibili:

New York
Londra
Tokyo
Sydney
Rio de Janeiro
Mosca
Pechino
Delhi
Berlino
Città del Messico
Johannesburg
Dubai
Singapore

Inserisci il nome della città che desideri per visualizzare l'orario attuale
(oppure scrivi 'esci' se desideri uscire dal programma): new york
L'ora attuale a New York è 05:21:15

Città disponibili:

New York
Londra
Tokyo
Sydney
Rio de Janeiro
Mosca
Pechino
Delhi
Berlino
Città del Messico
Johannesburg
Dubai
Singapore

Inserisci il nome della città che desideri per visualizzare l'orario attuale
(oppure scrivi 'esci' se desideri uscire dal programma): tokyo
L'ora attuale a Tokyo è 18:21:21

Città disponibili:

New York
Londra
Tokyo
Sydney
Rio de Janeiro
Mosca
Pechino
Delhi

Berlino
Città del Messico
Johannesburg
Dubai
Singapore

Inserisci il nome della città che desideri per visualizzare l'orario attuale
(oppure scrivi 'esci' se desideri uscire dal programma): LONDRA
L'ora attuale a Londra è 09:21:24

Città disponibili:

New York
Londra
Tokyo
Sydney
Rio de Janeiro
Mosca
Pechino
Delhi
Berlino
Città del Messico
Johannesburg
Dubai
Singapore

Inserisci il nome della città che desideri per visualizzare l'orario attuale
(oppure scrivi 'esci' se desideri uscire dal programma): SiNgApOrE
L'ora attuale a Singapore è 17:21:34

Città disponibili:

New York
Londra
Tokyo
Sydney
Rio de Janeiro
Mosca
Pechino
Delhi
Berlino
Città del Messico
Johannesburg
Dubai
Singapore

Inserisci il nome della città che desideri per visualizzare l'orario attuale
(oppure scrivi 'esci' se desideri uscire dal programma): prova

La città inserita non è nella lista. Per favore, prova di nuovo e controlla di aver scritto bene il nome della città (il nome deve essere scritto in maniera corretta per poter funzionare il programma).

Città disponibili:

New York
Londra
Tokyo
Sydney
Rio de Janeiro
Mosca
Pechino
Delhi
Berlino
Città del Messico
Johannesburg
Dubai
Singapore

Inserisci il nome della città che desideri per visualizzare l'orario attuale (oppure scrivi 'esci' se desideri uscire dal programma): esci
Arrivederci e grazie per aver usato questo programma!

23 IL CALCOLO DEL PESO BMI

Questo programma, chiamato “Calcolatore BMI”, consente agli utenti di calcolare e valutare l’Indice di Massa Corporea (BMI) per un numero specificato di persone.

```
[1]: def calcolabmi(peso,altezza):#definizione calcolo BMI
    return peso/(altezza**2)
def valutabmi(bmi):#definizione valutazione BMI
    if bmi<18.5:
        return "sottopeso"
    elif 18.5<=bmi<24.9:
        return "normopeso"
    elif 25<=bmi<29.9:
        return "sovrapeso"
    else:
        return "obeso"
def main():#definizione main
    print("Ciao, benvenuto nel calcolatore BMI!")
    print("Tramite questo programma puoi calcolare il BMI inserendo i dati\u2192necessari per il calcolo")
    numeropersone=int(input("Per prima cosa, inserisci il numero di persone di\u2192cui desideri calcolare il BMI: "))
    for i in range(numeropersone):
        peso=float(input("Inserisci il peso in chilogrammi: "))
```

```

altezza=float(input("Inserisci l'altezza in metri: "))
bmi=calcolabmi(peso, altezza)
valutazionefinale=valutabmi(bmi)
print(f"Il BMI è di {bmi:.2f} ed è classificato come {valutazionefinale}.")
→ ")
if __name__ == "__main__":
    main()

```

Ciao, benvenuto nel calcolatore BMI!

Tramite questo programma puoi calcolare il BMI inserendo i dati necessari per il calcolo

Per prima cosa, inserisci il numero di persone di cui desideri calcolare il BMI:
4

Inserisci il peso in chilogrammi: 60

Inserisci l'altezza in metri: 1.70

Il BMI è di 20.76 ed è classificato come normopeso.

Inserisci il peso in chilogrammi: 80

Inserisci l'altezza in metri: 1.75

Il BMI è di 26.12 ed è classificato come sovrappeso.

Inserisci il peso in chilogrammi: 70

Inserisci l'altezza in metri: 1.80

Il BMI è di 21.60 ed è classificato come normopeso.

Inserisci il peso in chilogrammi: 90

Inserisci l'altezza in metri: 1.65

Il BMI è di 33.06 ed è classificato come obeso.

24 LA FUNZIONE MAIN

La funzione Matteo() serve come punto di partenza per un programma Python, interagendo con l'utente per ricevere un input e confermando quindi che l'input è stato acquisito correttamente.

```
[2]: #Funzione principale che può avere qualsiasi nome (nome base: main)
def Matteo():#definisco Matteo
    print("Prova della funzione main, che ho chiamato in questo caso Matteo")
    →perchè posso chiamarla in qualsiasi modo")
    prova=(input("Prova a scrivere qualcosa dentro questa casella: "))
    print(f"Vedi che va perchè infatti tu hai scritto: {prova}")
    print("Quindi abbiamo visto che la funzione Matteo funziona perchè ha")
    →richiamato tutto il codice del programma")
if __name__=="__main__":
    Matteo()
#Il programma chiama la funzione "Matteo()" soltanto se il file è eseguito come
→script principale.
```

Prova della funzione main, che ho chiamato in questo caso Matteo perchè posso chiamarla in qualsiasi modo

Prova a scrivere qualcosa dentro questa casella: Ciao mi chiamo Matteo

Vedi che va perchè infatti tu hai scritto: Ciao mi chiamo Matteo

Quindi abbiamo visto che la funzione Matteo funziona perchè ha richiamato tutto il codice del programma

25 IL CONVERTITORE DI UNITÀ DI MISURA UNIVERSALE (CON LA FUNZIONE MAIN)

Questo programma è un convertitore di unità di misura universale che consente all'utente di convertire tra diverse unità di misura, tra cui metri, piedi, chilogrammi, libbre, centimetri e pollici.

```
[10]: def dametriapièdi(metri):
    return metri*3.28084
def dapièdiametri(piedi):
    return piedi/3.28084
def dachilogrammiallibbre(chilogrammi):
    return chilogrammi*2.20462
def dalibbrechilogrammi(libbre):
    return libbre/2.20462
def dapolicialicentimetri(pollici):
    return pollici*2.54
def dacentimetrialipollici(centimetri):
    return centimetri/2.54
def selezione(scelta):
    if scelta=="metri":
        valore=float(input("Inserisci il specifico valore in metri: "))
        risultato=dametriapièdi(valore)
        print(f"{valore:.3f} metri corrispondono a {risultato:.3f} piedi")
    elif scelta=="piedi":
        valore=float(input("Inserisci il specifico valore in piedi: "))
        risultato=dapièdiametri(valore)
        print(f"{valore} piedi corrispondono a {risultato} metri")
    elif scelta=="chilogrammi":
        valore=float(input("Inserisci il specifico valore in chilogrammi: "))
        risultato=dachilogrammiallibbre(valore)
        print(f"{valore} chilogrammi corrispondono a {risultato} libbre")
    elif scelta=="libbre":
        valore=float(input("Inserisci il specifico valore in libbre: "))
        risultato=dalibbrechilogrammi(valore)
        print(f"{valore} libbre corrispondono a {risultato} chilogrammi")
    elif scelta=="centimetri":
        valore=float(input("Inserisci il specifico valore in centimetri: "))
        risultato=dacentimetrialipollici(valore)
        print(f"{valore} centimetri corrispondono a {risultato} pollici")
    elif scelta=="pollici":
        valore=float(input("Inserisci il specifico valore in pollici: "))
        risultato=dapolicialicentimetri(valore)
        print(f"{valore} pollici corrispondono a {risultato} centimetri")
    else:
```

```

        print("Scelta non riconosciuta. Scegli tra una delle seguenti unità di misura riavviando prima il programma: metri/piedi/chilogrammi/libbre/centimetri/pollici")
def main():
    print("Ciao, benvenuto nel convertitore di unità di misura universale!")
    scelta=input("Quindi adesso scrivi quale unità di misura desideri convertire (puoi scegliere soltanto tra queste opzioni: metri/piedi/chilogrammi/libbre/centimetri/pollici): ").lower()
    selezione(scelta)
if __name__ == "__main__":
    main()

```

Ciao, benvenuto nel convertitore di unità di misura universale!
 Quindi adesso scrivi quale unità di misura desideri convertire (puoi scegliere soltanto tra queste opzioni: metri/piedi/chilogrammi/libbre/centimetri/pollici):
 POLLICI
 Inserisci il specifico valore in pollici: 17.3
 17.3 pollici corrispondono a 43.942 centimetri

26 IL CALCOLO DELLA CALORIE

Questo programma è un calcolatore di calorie consumate durante la giornata.

```
[3]: cibocalorie={
    "pizza": 285,
    "hamburger": 250,
    "insalata": 100,
    "pasta": 158,
    "pollo arrosto": 195,
    "riso": 130,
    "sushi": 374,
    "lasagna": 336,
    "fragole": 32,
    "gelato": 207,
    "panino": 320,
    "patatine fritte": 365,
    "uva": 69,
    "cioccolato": 546,
    "yogurt": 61,
    "spaghetti": 131,
    "pesce al vapore": 95,
    "muffin": 444,
    "cereali": 363,
    "torta al cioccolato": 237,
    "banane": 89,
    "popcorn": 365,
    "pollo fritto": 320,
```

"mele": 52,
"hot dog": 150,
"cavolfiore": 25,
"taco": 195,
"peperoni": 31,
"salsiccia": 229,
"cannella roll": 81,
"cetrioli": 16,
"maiale arrosto": 143,
"baguette": 299,
"frittata": 143,
"mela caramellata": 216,
"spiedini di pollo": 212,
"patate al forno": 161,
"pasticcio": 265,
"ananas": 50,
"lasagne al pesto": 320,
"ciambella": 190,
"sorbetto": 70,
"pepe": 3,
"pollo teriyaki": 250,
"ciambellone": 330,
"insalata di frutta": 74,
"pollo alla griglia": 165,
"croissant": 272,
"ramen": 186,
"fagioli neri": 132,
"pomodori": 18,
"toast al burro": 80,
"formaggio": 402,
"cavolo": 25,
"biscotti al cioccolato": 50,
"frutti di bosco": 32,
"gnocchi": 130,
"waffle": 266,
"anatra arrosto": 337,
"panna cotta": 366,
"tortellini": 181,
"panettone": 320,
"cioccolata calda": 192,
"insalata greca": 139,
"polpette": 320,
"tortilla": 297,
"patate fritte": 365,
"anelli di cipolla": 400,
"gelato alla vaniglia": 207,
"biscotti al burro": 496,

"bistecca": 250,
"zuppa di pomodoro": 74,
"insalata di pollo": 184,
"pollo tikka masala": 360,
"frappuccino": 250,
"uova strapazzate": 143,
"panino al tonno": 490,
"insalata caprese": 270,
"peperoni ripieni": 128,
"tiramisù": 370,
"pollo al limone": 225,
"cioccolato fondente": 604,
"anguria": 30,
"panini al formaggio": 314,
"caviale": 264,
"nachos": 364,
"cannella roll": 420,
"panino al prosciutto": 230,
"bistecca al pepe": 387,
"pollo al curry": 220,
"torta al limone": 326,
"tè dolce": 120,
"pollo alla senape": 210,
"muffin alle more": 377,
"pollo al pesto": 220,
"frittelle": 102,
"panini al pollo": 240,
"gelato alla fragola": 266,
"pasta al pesto": 400,
"fondue al formaggio": 249,
"patate al curry": 77,
"insalata di patate": 143,
"muffin al cioccolato": 444,
"insalata caesar": 184,
"panino al tacchino": 150,
"cioccolato al latte": 540,
"torta di carote": 237,
"ramen al pollo": 440,
"cheesecake": 321,
"panino al salmone": 300,
"pollo alla piastra": 177,
"cioccolato bianco": 540,
"torta di mele": 323,
"insalata di cetrioli": 45,
"hot dog al formaggio": 290,
"ciambella alla cannella": 253,
"panino vegetariano": 300,

```

    "pollo al peperoncino": 210,
}

def calorieconsumate(cibo, quantita):
    if cibo not in cibocalorie.keys():
        return -1
    else:
        return cibocalorie[cibo]*quantita

def main():
    ciboconsumato=[]
    print("Ciao, benvenuto nel calcolatore di calorie consumate della giornata!")
    print("Tramite questo programma puoi calcolare le calorie consumate inserendo i dati necessari per il calcolo")
    while True:
        print("Menù (selezionare un'opzione scrivendo il numero correlato): ")
        print("1. Aggiungi cibo consumato durante la giornata")
        print("2. Calcola calorie totali consumate durante la giornata")
        print("3. Esci dal programma")
        scelta=input("Scegli un'opzione: ")
        if scelta=="1":
            print("\nEcco l'elenco di cibi disponibili nel programma:")
            for key, value in cibocalorie.items():
                print(f"{key.capitalize()}: {value} calorie per 100g")
            cibo=input("Inserisci il cibo consumato: ").lower()
            quantita=float(input("Inserisci la quantità in grammi: "))
            calorie=calorieconsumate(cibo, quantita)
            if calorie==-1:
                print(f"Il cibo '{cibo}' non è presente nell'elenco. Inserisci un cibo valido ridezionando l'opzione 1 e scegli un cibo presente nell'elenco.")
            else:
                ciboconsumato.append((cibo, quantita))
        elif scelta=="2":
            calorietotali = sum(calorieconsumate(cibo, quantita) for cibo, quantita in ciboconsumato)
            print(f"Hai consumato un totale di {calorietotali} calorie.")
        elif scelta=="3":
            print("Arrivederci e grazie per aver usato questo programma!")
            break
        else:
            print("Scelta non valida. Riprova. Puoi scegliere solo una delle tre opzioni (1, 2, 3)")

if __name__ == "__main__":
    main()

```

Ciao, benvenuto nel calcolatore di calorie consumate della giornata!
Tramite questo programma puoi calcolare le calorie consumate inserendo i dati necessari per il calcolo

Menù (selezionare un'opzione scrivendo il numero correlato):

1. Aggiungi cibo consumato durante la giornata
2. Calcola calorie totali consumate durante la giornata
3. Esci dal programma

Scegli un'opzione: 1

Ecco l'elenco di cibi disponibili nel programma:

Pizza: 285 calorie per 100g
Hamburger: 250 calorie per 100g
Insalata: 100 calorie per 100g
Pasta: 158 calorie per 100g
Pollo arrosto: 195 calorie per 100g
Riso: 130 calorie per 100g
Sushi: 374 calorie per 100g
Lasagna: 336 calorie per 100g
Fragole: 32 calorie per 100g
Gelato: 207 calorie per 100g
Panino: 320 calorie per 100g
Patatine fritte: 365 calorie per 100g
Uva: 69 calorie per 100g
Cioccolato: 546 calorie per 100g
Yogurt: 61 calorie per 100g
Spaghetti: 131 calorie per 100g
Pesce al vapore: 95 calorie per 100g
Muffin: 444 calorie per 100g
Cereali: 363 calorie per 100g
Torta al cioccolato: 237 calorie per 100g
Banane: 89 calorie per 100g
Popcorn: 365 calorie per 100g
Pollo fritto: 320 calorie per 100g
Mele: 52 calorie per 100g
Hot dog: 150 calorie per 100g
Cavolfiore: 25 calorie per 100g
Taco: 195 calorie per 100g
Peperoni: 31 calorie per 100g
Salsiccia: 229 calorie per 100g
Cannella roll: 420 calorie per 100g
Cetrioli: 16 calorie per 100g
Maiale arrosto: 143 calorie per 100g
Baguette: 299 calorie per 100g
Frittata: 143 calorie per 100g
Mela caramellata: 216 calorie per 100g
Spiedini di pollo: 212 calorie per 100g
Patate al forno: 161 calorie per 100g
Pasticcio: 265 calorie per 100g
Ananas: 50 calorie per 100g
Lasagne al pesto: 320 calorie per 100g
Ciambella: 190 calorie per 100g

Sorbetto: 70 calorie per 100g
Pepe: 3 calorie per 100g
Pollo teriyaki: 250 calorie per 100g
Ciambellone: 330 calorie per 100g
Insalata di frutta: 74 calorie per 100g
Pollo alla griglia: 165 calorie per 100g
Croissant: 272 calorie per 100g
Ramen: 186 calorie per 100g
Fagioli neri: 132 calorie per 100g
Pomodori: 18 calorie per 100g
Toast al burro: 80 calorie per 100g
Formaggio: 402 calorie per 100g
Cavolo: 25 calorie per 100g
Biscotti al cioccolato: 50 calorie per 100g
Frutti di bosco: 32 calorie per 100g
Gnocchi: 130 calorie per 100g
Waffle: 266 calorie per 100g
Anatra arrosto: 337 calorie per 100g
Panna cotta: 366 calorie per 100g
Tortellini: 181 calorie per 100g
Panettone: 320 calorie per 100g
Cioccolata calda: 192 calorie per 100g
Insalata greca: 139 calorie per 100g
Polpette: 320 calorie per 100g
Tortilla: 297 calorie per 100g
Patate fritte: 365 calorie per 100g
Anelli di cipolla: 400 calorie per 100g
Gelato alla vaniglia: 207 calorie per 100g
Biscotti al burro: 496 calorie per 100g
Bistecca: 250 calorie per 100g
Zuppa di pomodoro: 74 calorie per 100g
Insalata di pollo: 184 calorie per 100g
Pollo tikka masala: 360 calorie per 100g
Frappuccino: 250 calorie per 100g
Uova strapazzate: 143 calorie per 100g
Panino al tonno: 490 calorie per 100g
Insalata caprese: 270 calorie per 100g
Peperoni ripieni: 128 calorie per 100g
Tiramisù: 370 calorie per 100g
Pollo al limone: 225 calorie per 100g
Cioccolato fondente: 604 calorie per 100g
Anguria: 30 calorie per 100g
Panini al formaggio: 314 calorie per 100g
Caviale: 264 calorie per 100g
Nachos: 364 calorie per 100g
Panino al prosciutto: 230 calorie per 100g
Bistecca al pepe: 387 calorie per 100g
Pollo al curry: 220 calorie per 100g

Torta al limone: 326 calorie per 100g
Tè dolce: 120 calorie per 100g
Pollo alla senape: 210 calorie per 100g
Muffin alle more: 377 calorie per 100g
Pollo al pesto: 220 calorie per 100g
Frittelle: 102 calorie per 100g
Panini al pollo: 240 calorie per 100g
Gelato alla fragola: 266 calorie per 100g
Pasta al pesto: 400 calorie per 100g
Fondue al formaggio: 249 calorie per 100g
Patate al curry: 77 calorie per 100g
Insalata di patate: 143 calorie per 100g
Muffin al cioccolato: 444 calorie per 100g
Insalata caesar: 184 calorie per 100g
Panino al tacchino: 150 calorie per 100g
Cioccolato al latte: 540 calorie per 100g
Torta di carote: 237 calorie per 100g
Ramen al pollo: 440 calorie per 100g
Cheesecake: 321 calorie per 100g
Panino al salmone: 300 calorie per 100g
Pollo alla piastra: 177 calorie per 100g
Cioccolato bianco: 540 calorie per 100g
Torta di mele: 323 calorie per 100g
Insalata di cetrioli: 45 calorie per 100g
Hot dog al formaggio: 290 calorie per 100g
Ciambella alla cannella: 253 calorie per 100g
Panino vegetariano: 300 calorie per 100g
Pollo al peperoncino: 210 calorie per 100g
Inserisci il cibo consumato: Panino al salmone
Inserisci la quantità in grammi: 555
Menù (selezionare un'opzione scrivendo il numero correlato):
1. Aggiungi cibo consumato durante la giornata
2. Calcola calorie totali consumate durante la giornata
3. Esci dal programma
Scegli un'opzione: 1

Ecco l'elenco di cibi disponibili nel programma:
Pizza: 285 calorie per 100g
Hamburger: 250 calorie per 100g
Insalata: 100 calorie per 100g
Pasta: 158 calorie per 100g
Pollo arrosto: 195 calorie per 100g
Riso: 130 calorie per 100g
Sushi: 374 calorie per 100g
Lasagna: 336 calorie per 100g
Fragole: 32 calorie per 100g
Gelato: 207 calorie per 100g
Panino: 320 calorie per 100g

Patatine fritte: 365 calorie per 100g
Uva: 69 calorie per 100g
Cioccolato: 546 calorie per 100g
Yogurt: 61 calorie per 100g
Spaghetti: 131 calorie per 100g
Pesce al vapore: 95 calorie per 100g
Muffin: 444 calorie per 100g
Cereali: 363 calorie per 100g
Torta al cioccolato: 237 calorie per 100g
Banane: 89 calorie per 100g
Popcorn: 365 calorie per 100g
Pollo fritto: 320 calorie per 100g
Mele: 52 calorie per 100g
Hot dog: 150 calorie per 100g
Cavolfiore: 25 calorie per 100g
Taco: 195 calorie per 100g
Peperoni: 31 calorie per 100g
Salsiccia: 229 calorie per 100g
Cannella roll: 420 calorie per 100g
Cetrioli: 16 calorie per 100g
Maiale arrosto: 143 calorie per 100g
Baguette: 299 calorie per 100g
Frittata: 143 calorie per 100g
Mela caramellata: 216 calorie per 100g
Spiedini di pollo: 212 calorie per 100g
Patate al forno: 161 calorie per 100g
Pasticcio: 265 calorie per 100g
Ananas: 50 calorie per 100g
Lasagne al pesto: 320 calorie per 100g
Ciambella: 190 calorie per 100g
Sorbetto: 70 calorie per 100g
Pepe: 3 calorie per 100g
Pollo teriyaki: 250 calorie per 100g
Ciambellone: 330 calorie per 100g
Insalata di frutta: 74 calorie per 100g
Pollo alla griglia: 165 calorie per 100g
Croissant: 272 calorie per 100g
Ramen: 186 calorie per 100g
Fagioli neri: 132 calorie per 100g
Pomodori: 18 calorie per 100g
Toast al burro: 80 calorie per 100g
Formaggio: 402 calorie per 100g
Cavolo: 25 calorie per 100g
Biscotti al cioccolato: 50 calorie per 100g
Frutti di bosco: 32 calorie per 100g
Gnocchi: 130 calorie per 100g
Waffle: 266 calorie per 100g
Anatra arrosto: 337 calorie per 100g

Panna cotta: 366 calorie per 100g
Tortellini: 181 calorie per 100g
Panettone: 320 calorie per 100g
Cioccolata calda: 192 calorie per 100g
Insalata greca: 139 calorie per 100g
Polpette: 320 calorie per 100g
Tortilla: 297 calorie per 100g
Patate fritte: 365 calorie per 100g
Anelli di cipolla: 400 calorie per 100g
Gelato alla vaniglia: 207 calorie per 100g
Biscotti al burro: 496 calorie per 100g
Bistecca: 250 calorie per 100g
Zuppa di pomodoro: 74 calorie per 100g
Insalata di pollo: 184 calorie per 100g
Pollo tikka masala: 360 calorie per 100g
Frappuccino: 250 calorie per 100g
Uova strapazzate: 143 calorie per 100g
Panino al tonno: 490 calorie per 100g
Insalata caprese: 270 calorie per 100g
Peperoni ripieni: 128 calorie per 100g
Tiramisù: 370 calorie per 100g
Pollo al limone: 225 calorie per 100g
Cioccolato fondente: 604 calorie per 100g
Anguria: 30 calorie per 100g
Panini al formaggio: 314 calorie per 100g
Caviale: 264 calorie per 100g
Nachos: 364 calorie per 100g
Panino al prosciutto: 230 calorie per 100g
Bistecca al pepe: 387 calorie per 100g
Pollo al curry: 220 calorie per 100g
Torta al limone: 326 calorie per 100g
Tè dolce: 120 calorie per 100g
Pollo alla senape: 210 calorie per 100g
Muffin alle more: 377 calorie per 100g
Pollo al pesto: 220 calorie per 100g
Frittelle: 102 calorie per 100g
Panini al pollo: 240 calorie per 100g
Gelato alla fragola: 266 calorie per 100g
Pasta al pesto: 400 calorie per 100g
Fondue al formaggio: 249 calorie per 100g
Patate al curry: 77 calorie per 100g
Insalata di patate: 143 calorie per 100g
Muffin al cioccolato: 444 calorie per 100g
Insalata caesar: 184 calorie per 100g
Panino al tacchino: 150 calorie per 100g
Cioccolato al latte: 540 calorie per 100g
Torta di carote: 237 calorie per 100g
Ramen al pollo: 440 calorie per 100g

Cheesecake: 321 calorie per 100g
 Panino al salmone: 300 calorie per 100g
 Pollo alla piastra: 177 calorie per 100g
 Cioccolato bianco: 540 calorie per 100g
 Torta di mele: 323 calorie per 100g
 Insalata di cetrioli: 45 calorie per 100g
 Hot dog al formaggio: 290 calorie per 100g
 Ciambella alla cannella: 253 calorie per 100g
 Panino vegetariano: 300 calorie per 100g
 Pollo al peperoncino: 210 calorie per 100g
 Inserisci il cibo consumato: dssssasfà
 Inserisci la quantità in grammi: 4554
 Il cibo 'dssssasfà' non è presente nell'elenco. Inserisci un cibo valido
 riselezionando l'opzione 1 e scegli un cibo presente nell'elenco.
 Menù (selezionare un'opzione scrivendo il numero correlato):
 1. Aggiungi cibo consumato durante la giornata
 2. Calcola calorie totali consumate durante la giornata
 3. Esci dal programma
 Scegli un'opzione: 21
 Scelta non valida. Riprova. Puoi scegliere solo una delle tre opzioni (1, 2, 3)
 Menù (selezionare un'opzione scrivendo il numero correlato):
 1. Aggiungi cibo consumato durante la giornata
 2. Calcola calorie totali consumate durante la giornata
 3. Esci dal programma
 Scegli un'opzione: 2
 Hai consumato un totale di 166500.0 calorie.
 Menù (selezionare un'opzione scrivendo il numero correlato):
 1. Aggiungi cibo consumato durante la giornata
 2. Calcola calorie totali consumate durante la giornata
 3. Esci dal programma
 Scegli un'opzione: 3
 Arrivederci e grazie per aver usato questo programma!

27 LA FUNZIONE TUPLA

Il codice crea una tupla chiamata lamiabellatupla con tre elementi di tipo stringa: "John", "Mark" e "Vicky". Successivamente, utilizza il metodo join() per unire gli elementi della tupla in una singola stringa, separati da virgolette. Infine, stampa il risultato della concatenazione e il tipo della variabile risultante.

```
[2]: #Creiamo una tupla 'lamiabellatupla' con tre elementi di tipo stringa di testo
      ↪(si può sempre chiamare come si vuole)
#Una tupla è una struttura dati immutabile in Python, simile a una lista ma non
      ↪può essere modificata dopo la creazione (mentre una lista sì).
lamiabellatupla=("John", " Mark", " Vicky")
#Usiamo 'join()' per unire gli elementi della tupla in una sola stringa,
      ↪separati da virgolette
```

```

x=",".join(lamiabellatupla)
#Stampiamo il risultato e il tipo della variabile 'x' come str (stringa di testo).
print("Prova della funzione tupla: ")
print(x, type(x))
print("Quindi la tupla funziona correttamente")

```

Prova della funzione tupla:
John, Mark, Vicy <class 'str'>
Quindi la tupla funziona correttamente

28 GLI USI DELLA FUNZIONE TUPLA

Questo codice dimostra l'utilizzo delle tuple in Python.

```

[12]: lamiabellatupla="Prova messaggio della tupla:"
print(lamiabellatupla)
lamiabellatupla=(1, 2, 3, "prova... prova, mi sentite tutti vero?")
print(lamiabellatupla)
lamiabellatupla="Ciao sono la funzione tupla, piacere di conoscerti!"
print(lamiabellatupla)
print("Ecco un esempio più approfondito di uso di tupla: ")
#Creazione di una tupla per rappresentare una persona con nome, età e indirizzo
personaesempio=("Alice", 30, "123 Main Street")
#Accesso ai dati nella tupla
nome, età, indirizzo=personaesempio
#Stampa dei dati
print("Il testo inserito nella tupla è: Alice, 30, 123 Main Street")
print("Nome:", nome)
print("Età:", età)
print("Indirizzo:", indirizzo)
print("Visto che utilità? La tupla è una struttura dati immutabile che consente di raggruppare dati correlati in una singola entità. Una volta creata, non è possibile modificarla se non ridefinendola, il che può essere utile per evitare errori nel codice e garantire che i dati rimangano consistenti")

```

Prova messaggio della tupla:
(1, 2, 3, 'prova... prova, mi sentite tutti vero?')
Ciao sono la funzione tupla, piacere di conoscerti!
Ecco un esempio più approfondito di uso di tupla:
Il testo inserito nella tupla è: Alice, 30, 123 Main Street
Nome: Alice
Età: 30
Indirizzo: 123 Main Street
Visto che utilità? La tupla è una struttura dati immutabile che consente di raggruppare dati correlati in una singola entità. Una volta creata, non è possibile modificarla se non ridefinendola, il che può essere utile per evitare errori nel codice e garantire che i dati rimangano consistenti

29 IL GENERATORE DI PERSONAGGI FANTASY

Questo script Python genera casualmente le caratteristiche di un personaggio fantasy, incluse specie, classe, arma e abilità

```
[23]: import random
speci=[ "Umano", "Elfo", "Nano", "Orco"] + [ "Goblin", "Drago", "Fata", "Licantropo"] + [ "Gigante", "Vampiro", "Stregone", "Fantasma"] + [ "Gnomo", "Mummia", "Sirena", "Satiro"] + [ "Elementale", "Angelo", "Demone", "Centauro"] + [ "Gorgone", "Guerriero delle ombre", "Ladro delle stelle", "Mago delle illusioni"] + [ "Spettro", "Orco magico", "Nano meccanico", "Elfo oscuro"] + [ "Incantatore", "Cacciatore di mostri", "Mago guerriero", "Lupo mannaro"] + [ "Stregone", "Custode della foresta", "Custode delle tombe", "Drago di ghiaccio"] + [ "Elementale del fuoco", "Chierico della luce", "Furia degli abissi", "Assassino dell'ombra"]
classi=[ "Guerriero", "Mago", "Ladro", "Chierico"] + [ "Bardo", "Paladino", "Cacciatore", "Necromante"] + [ "Barbaro", "Ingegnere", "Assassino", "Sciamano"] + [ "Birifrangente", "Avventuriero", "Monaco", "Arciere"] + [ "Bardo oscuro", "Inventore", "Maestro delle trappole", "Custode della magia"] + [ "Sovrano delle ombre", "Cavaliere sacro", "Maestro delle bestie", "Mago dell'illusione"] + [ "Signore della guerra", "Alchimista", "Agente segreto", "Guardiano del tempio"] + [ "Custode della natura", "Custode delle anime", "Maestro delle maledizioni", "Signore dei venti"] + [ "Mastro d'armi", "Esploratore", "Predatore di draghi", "Cacciatore di streghe"] + [ "Distruttore di incantesimi", "Capo tribù", "Mago della cripta", "Signore dei ghiacci"]
armi=[ "Spada", "Arco", "Bastone", "Pugnale"] + [ "Martello", "Lancia", "Fionda", "Mazza"] + [ "Arco magico", "Daga avvelenata", "Scettro magico", "Balestra"] + [ "Ascia bifronte", "Fulmine a catena", "Lama dell'ombra", "Frusta acida"] + [ "Lama incantata", "Arco di cristallo", "Bastone della guarigione", "Daga del vento"] + [ "Martello della lava", "Lancia avvelenata", "Fionda magica", "Mazza dell'oblio"] + [ "Arco delle stelle", "Daga dell'illusione", "Scettro dell'incantesimo", "Balestra delle ombre"] + [ "Ascia delle tempeste", "Fulmine a catena infuocato", "Lama dell'oscurità", "Frusta velenosa"] + [ "Lama ghiacciata", "Arco dell'etere", "Bastone della saggezza", "Pugnale del destino"] + [ "Ascia del caos", "Fulmine a catena sacro", "Lama dell'equilibrio", "Frusta della perdizione"]
```

```

abilita=["Fuoco", "Gelo", "Cura", "Invisibilità", "Forza", "Velocità"] +_
→["Teletrasporto", "Controllo mentale", "Rigenerazione", "Illusione"] +_
→["Esplosione magica", "Paralisi", "Proiezione astrale", "Trasformazione"] +_
→["Rallentamento del tempo", "Telecinesi", "Manipolazione elementale",_
→"Camuffamento"] + ["Telepatia", "Assorbimento vitale", "Scudo magico",_
→"Esplosione psichica"] + ["Rianimazione", "Teletrasporto interdimensionale",_
→"Rigenerazione accelerata", "Invisibilità permanente"] + ["Controllo delle_
→creature", "Telecinesi avanzata", "Illusione collettiva", "Assorbimento_
→energetico"] + ["Manipolazione temporale", "Teletrasporto istantaneo", "Cura_
→stantanea", "Invisibilità totale"] + ["Evocazione di creature", "Controllo_
→totale delle menti", "Rigenerazione istantanea", "Illusione suprema"] +_
→["Distorsione della realtà", "Teletrasporto galattico", "Manipolazione della_
→vita", "Assorbimento cosmico"]

specie=random.choice(speci)
classe=random.choice(classi)
arma=random.choice(armi)

abilitas=random.sample(abilita, random.randint(1, 3))

print("Ciao, benvenuto nel generatore di personaggi Fantasy")
print("Tramite questo programma puoi generare dei personaggi fantasy in maniera_
→del tutto casuale")
print("Il personaggio fantasy generato è: ")
print(f"Specie: {specie}")
print(f"Classe: {classe}")
print(f"Arma: {arma}")
print(f"Abilità: {', '.join(abilitas)}")

```

Ciao, benvenuto nel generatore di personaggi Fantasy
Tramite questo programma puoi generare dei personaggi fantasy in maniera del tutto casuale
Il personaggio fantasy generato è:
Specie: Gnomo
Classe: Maestro delle trappole
Arma: Scettro dell'incantesimo
Abilità: Assorbimento cosmico

30 IL GENERATORE DI PERSONAGGI FANTASY (CON LA FUNZIONE MAIN)

Questo script Python genera casualmente le caratteristiche di un personaggio fantasy, inclusi aspetto fisico, aspetto personale, sfondo sociale e motivazione.

```
[4]: import random
trattifisiomatici=["Capelli neri", "Capelli biondi", "Occhi azzurri", "Alto",_
→"Basso", "Barba lunga", "Capelli rossi", "Capelli grigi", "Occhi verdi",_
→"Vestiti eleganti", "Vestiti trasandati", "Senza barba", "Capelli ricci",_
→"Capelli lisci", "Occhi marroni", "Atletico", "Corpulento", "Barba corta",_
→"Capelli corti", "Occhi grigi",
```

"Capelli viola", "Occhi celesti", "Pelle scura", "Pelle chiara", "Capelli
 →rosa", "Capelli verdi", "Barba incolta", "Capelli lunghi", "Occhi arancioni",
 →"Vestiti colorati", "Vestiti scuri", "Capelli blu", "Occhi viola", "Pelle
 →rugosa", "Capelli argento", "Occhi dorati", "Capelli rasta", "Pelle tatuata",
 →"Capelli grigi", "Occhi ambrati", "Capelli multicolori",
 "Pelle cicatrizzata", "Occhi senza pupilla", "Capelli incolti", "Occhi
 →luminosi", "Pelle trasparente", "Occhi a mandorla", "Capelli ondulati",
 →"Capelli rasati", "Occhi strabici", "Vestiti stravaganti", "Vestiti
 →minimalisti", "Pelle iridescente", "Occhi di vetro", "Capelli crespi", "Occhi
 →spenti", "Capelli folti", "Capelli sottili", "Pelle vellutata", "Occhi
 →incavati"]
 trattipsicologici=["Gentile", "Arrogante", "Timido", "Audace", "Curioso",
 →"Generoso", "Introverso", "Estroverso", "Misterioso", "Sensibile", "Paziente",
 →"Impulsivo", "Riflessivo", "Spontaneo", "Ottimista", "Pessimista", "Leale",
 →"Ingnanveole", "Ambizioso", "Rilassato",
 "Empatico", "Egoista", "Razionale", "Emotivo", "Altruista", "Individualista",
 →"Indeciso", "Deciso", "Realista", "Idealista", "Sicuro di sé", "Inceto",
 →"Socievole", "Solitario", "Flessibile", "Rigido", "Collaborativo",
 →"Competitivo", "Avventuroso", "Cauto", "Innovativo",
 "Tradizionalista", "Umile", "Presuntuoso", "Ottimista", "Malinconico",
 →"Allegro", "Serio", "Hobbyista", "Professionale", "Sensato", "Stravagante",
 →"Scettico", "Credulone", "Orientato all'obiettivo", "Orientato al processo",
 →"Conformista", "Ribelle", "Pigrone", "Determinato",
 "Adattabile", "Testardo", "Amichevole", "Scontroso", "Entusiasta",
 →"Indifferente", "Accogliente", "Freddo", "Stoico", "Impulsivo", "Metodico",
 →"Caotico", "Organizzato", "Competente", "Incompetente", "Grato", "Rancoroso",
 →"Generoso", "Avaro", "Sensato", "Impulsivo"]
 sfondisociali=["Contadino", "Nobile", "Guerriero", "Mercante", "Mago",
 →"Artigiano", "Esploratore", "Ladro", "Scholarly", "Pirata", "Cavaliere",
 →"Sacerdote", "Mercenario", "Commercante", "Stregone", "Ingegnere", "Spia",
 →"Accademico", "Nauta", "Alchimista",
 "Furfante", "Sovrano", "Guardiano", "Rivoluzionario", "Musicista", "Colono",
 →"Custode", "Rinnegato", "Poeta", "Cavaliere errante", "Inquisitore",
 →"Cacciatore di taglie", "Sopravvissuto", "Ambasciatore", "Predicatore",
 →"Prigioniero", "Contadino povero", "Profeta", "Guaritore", "Disertore",
 "Esule", "Guardia cittadina", "Mercenario senza scrupoli", "Scienziato pazzo",
 →"Guerrigliero", "Cavallerizzo", "Taglialegna", "Cantastorie", "Astronomo",
 →"Eremita", "Cacciatore", "Custode della biblioteca", "Acrobata", "Fabbro",
 →"Ammaliatore", "Spadaccino", "Perito marittimo", "Cartografo", "Diplomatico",
 "Custode delle tombe", "Maestro di armi", "Banchiere corrotto", "Custode del
 →sapere", "Guerriero della luce", "Assassino", "Cittadino medio", "Barista",
 →"Mago oscuro", "Capo di una gilda", "Guida spirituale", "Capo di una fazione",
 →"Cavaliere senza paura", "Combattente dell'arena", "Collezionista d'arte",
 →"Custode degli animali", "Ladro d'arte"]

```

motivazioni=[ "Vendetta", "Ricchezza", "Potere", "Amore", "Scoperta", "Giustizia", "Vivere l'avventura", "Vengeance", "Redenzione", "Fama", "Salute", "Conoscenza", "Libertà", "Creatività", "Amicizia", "Avventura", "Sopravvivenza", "Eccellenza", "Competizione", "Armonia", "Ricerca della verità", "Gioia dell'esplorazione", "Curiosità scientifica", "Desiderio di cambiamento", "Nostalgia", "Desiderio di conquista", "Crescita personale", "Desiderio di pace", "Riparazione di errori", "Esplorazione dell'ignoto", "Superare le sfide", "Apprendimento continuo", "Aiutare gli altri", "Risolvere misteri", "Creare qualcosa di duraturo", "Sfida personale", "Influenzare il mondo", "Proteggere gli innocenti", "Avere successo", "Incontrare nuove persone", "Avere una famiglia", "Esprimere sé stessi", "Divertirsi", "Sperimentare nuove culture", "Avere potere", "Ritornare a casa", "Essere accettato", "Sconfiggere il male", "Esplorare l'arte", "Comprendere la mente umana", "Dominare la magia", "Diventare una leggenda", "Connettersi con la natura", "Superare le proprie paure", "Essere indipendenti", "Superare le avversità", "Ripristinare l'equilibrio", "Affrontare il destino", "Scoprire segreti antichi"]
def generatorepersonaggi():
    nome=input("Per prima cosa scegli il nome con cui vorresti chiamare il personaggio generato: ")
    print("Il personaggio fantasy generato è: ")
    aspettofisico=random.choice(trattifisiomatici)
    aspettopersonale=random.choice(trattipsicologici)
    sfondosociale=random.choice(sfondisociali)
    motivazione=random.choice(motivazioni)
    descrizione=(
        f"Nome: {nome}\n"
        f"Aspetto fisico: {aspettofisico}\n"
        f"Aspetto personale: {aspettopersonale}\n"
        f"Sfondo sociale: {sfondosociale}\n"
        f"Motivazione: {motivazione}"
    )
    return descrizione
def main():
    print("Ciao, benvenuto nel generatore di personaggi Fantasy")
    print("Tramite questo programma puoi generare dei personaggi fantasy in maniera del tutto casuale")
    print(generatorepersonaggi())
if __name__ == "__main__":
    main()

```

Ciao, benvenuto nel generatore di personaggi Fantasy
Tramite questo programma puoi generare dei personaggi fantasy in maniera del tutto casuale
Per prima cosa scegli il nome con cui vorresti chiamare il personaggio generato:
Matteo
Il personaggio fantasy generato è:

Nome: Matteo
Aspetto fisico: Capelli multicolori
Aspetto personale: Conformista
Sfondo sociale: Accademico
Motivazione: Affrontare il destino

31 IL GENERATORE DI CITAZIONI DEL GIORNO (CON LA FUNZIONE MAIN)

Questo script Python genera casualmente una citazione del giorno da una lista di citazioni predefinite.

```
[51]: def genera_citazione_del_giorno():
    #Lista di 100 citazioni del giorno
    citazioni = [
        "Il futuro appartiene a coloro che credono nella bellezza dei propri sogni. - Eleanor Roosevelt",
        "Il successo è camminare da un fallimento all'altro senza perdere l'entusiasmo. - Winston Churchill",
        "La vita è 10% ciò che ci accade e 90% come reagiamo ad essa. - Charles R. Swindoll",
        "La mente è tutto. Tu diventi ciò che pensi. - Buddha",
        "Non importa quanti passi fai indietro, a condizione che ne faccia uno in avanti. - Abraham Lincoln",
        "La vita è ciò che accade mentre sei occupato a fare altri progetti. - John Lennon",
        "Il modo in cui inizi la giornata determina come si svolgerà. - Robin Sharma",
        "La pazienza è amara, ma il suo frutto è dolce. - Jean-Jacques Rousseau",
        "Il successo non è la chiave della felicità. La felicità è la chiave del successo. - Albert Schweitzer",
        "Sii il cambiamento che vuoi vedere nel mondo. - Mahatma Gandhi",
        "La tua vita migliorerà cambiando le tue abitudini. - Unknown",
        "Il fallimento è l'opportunità di ricominciare con maggiore intelligenza. - Henry Ford",
        "La saggezza è sapere che cosa fare; la virtù è farlo. - David Starr Jordan",
        "Il modo migliore per prevedere il futuro è crearlo. - Peter Drucker",
        "Il tuo tempo è limitato, non sprecarlo vivendo la vita di qualcun altro. - Steve Jobs",
        "La felicità è una scelta, non un risultato. - Ralph Marston",
        "Il successo è ottenere ciò che vuoi. La felicità è volere ciò che ottieni. - Dale Carnegie",
        "Non è mai troppo tardi per essere ciò che avresti potuto essere. - George Eliot",
```

"L'unica limitazione che hai è quella che tu stai imponendo a te stesso. -
→- Dr. Wayne Dyer",
"Se vuoi raggiungere il picco, devi superare la cresta. - William S. Burroughs",
"L'azione è la chiave fondamentale di ogni successo. - Pablo Picasso",
"Il modo in cui si inizia la giornata determina come si sviluppa. - Robin Sharma",
"La vera opportunità per il successo risiede nella persona e non nelle circostanze. - Zig Ziglar",
"Ogni giorno è una nuova opportunità per fare qualcosa di straordinario. - Unknown",
"Non aspettare che le circostanze siano a tuo favore. Crea le circostanze. - George Bernard Shaw",
"L'unica cosa che si frappone tra te e il tuo obiettivo è la storia che ti racconti su perché non puoi raggiungerlo. - Jordan Belfort",
"La tua vita cambierà solo quando tu cambierai. - Jim Rohn",
"Il segreto per ottenere ciò che vuoi è chiedere. - Jim Rohn",
"La tua paura più profonda non è che tu sia inadeguato. La tua paura più profonda è che tu sia potente oltre ogni misura. - Marianne Williamson",
"Le piccole azioni, ripetute giorno dopo giorno, portano al successo. - Unknown",
"L'unico modo per fare un grande lavoro è amare quello che fai. - Steve Jobs",
"Il cambiamento è l'unica costante nella vita. - Heraclitus"
"La vita è come una bicicletta: per mantenere l'equilibrio, devi muoverti in avanti. - Albert Einstein",
"Il successo non è la fine, il fallimento non è fatale: è il coraggio di continuare che conta. - Winston Churchill",
"La felicità non è qualcosa fatto. Dipende dalle tue azioni. - Dalai Lama",
"Se vuoi volare, devi liberarti dal peso che ti trascina verso il basso. - Toni Morrison",
"Il segreto del successo è imparare come usare il dolore e godere del lavoro. - A.P.J. Abdul Kalam",
"La vita è troppo importante per essere presa sul serio. - Oscar Wilde",
"Non si tratta di quanto colpisci, ma di quanto colpisci e continui a muoverti in avanti. - Rocky Balboa",
"L'unico modo per fare un grande lavoro è amare quello che fai. - Steve Jobs",
"Non cercare la colpa. Cerca una soluzione. - Henry Ford",
"La vita è fatta di sogni e di emozioni. Vivi la tua vita al massimo. - Unknown",
"Il coraggio non è l'assenza di paura, ma il trionfo su di essa. - Nelson Mandela",
"La creatività è contagiosa, trasmettila. - Albert Einstein",

"Il successo è la somma di piccoli sforzi ripetuti giorno dopo giorno. -
→Robert Collier",

"La perseveranza non è una corsa lunga; è molti sprints corti, uno dopo
→l'altro. - Walter Elliot",

"Il segreto del cambiamento è concentrarsi su costruire il nuovo, non
→combattere l'antico. - Socrate",

"La vita è troppo breve per spendere il tuo tempo prezioso cercando di
→convincere una persona che non è disposta ad ascoltarti. - Shannon L. Alder",

"Sii selettivo nelle tue battaglie, a volte avere la pace è meglio che
→avere ragione. - Unknown",

"Il fallimento è l'opportunità di iniziare di nuovo, con maggiore
→saggezza. - Catherine Pulsifer",

"La vita è ciò che accade quando sei occupato a fare altri progetti. -
→Allen Sanders",

"Il modo in cui affronti le sfide determina il tuo successo. - Unknown",

"Ogni giorno è un'opportunità per un nuovo inizio. - Unknown",

"La felicità è un'abilità che si sviluppa e si pratica, non qualcosa che
→trovi. - Ricard Carlson",

"La tua mente è un giardino. I tuoi pensieri sono i semi. Puoi crescere
→fiori o puoi crescere erbacce. - Unknown",

"Il successo è come un iceberg. Molte persone vedono solo la punta, ma
→non conoscono il lavoro svolto sotto la superficie. - Unknown",

"Il tuo tempo è limitato, non sprecarlo vivendo la vita di qualcun altro.
→ - Unknown",

"Il progresso non è mai un risultato accidentale, ma una regola. -
→Philip J. Romano",

"Il modo migliore per predire il futuro è crearlo. - Peter Drucker",

"La tua attitudine determina la tua direzione. - Unknown",

"Il successo è ottenere ciò che vuoi. La felicità è volere ciò che
→ottieni. - W.P. Kinsella",

"Non guardare indietro con rimpianto, guarda avanti con speranza. -
→Unknown"

"La vita è un'avventura audace o niente. - Helen Keller",

"La felicità è la chiave del successo. Se ami ciò che fai, avrai
→successo. - Albert Schweitzer",

"La tua vita migliorerà solo quando tu lo farai. - Unknown",

"Le sfide sono ciò che rendono la vita interessante e superarle è ciò
→che la rende significativa. - Joshua J. Marine",

"Il successo è la somma di piccoli sforzi ripetuti giorno dopo giorno. -
→Robert Collier",

"La vita è fatta di momenti, non aspettare che passino, crea quelli che
→desideri. - Unknown",

"La tua attitudine determina la tua altitudine. - Zig Ziglar",

"Il fallimento è solo l'opportunità di iniziare di nuovo, questa volta
→in modo più intelligente. - Henry Ford",

"Il progresso è impossibile senza cambiamento, e coloro che non possono cambiare la loro mente non possono cambiare nulla. - George Bernard Shaw",

"Le grandi menti discutono idee, le menti medie discutono eventi, le menti piccole discutono persone. - Eleanor Roosevelt",

"La tua vita è il tuo messaggio al mondo. Assicurati che sia un buon messaggio. - Unknown",

"Il segreto della felicità è la libertà, il segreto della libertà è il coraggio. - Carrie Jones",

"La vita è troppo corta per essere infelice. Quindi smetti di ascoltare le voci che dicono che non puoi e inizia a fare ciò che ami. - Unknown",

"La vera saggezza è imparare dalle esperienze degli altri. - Unknown",

"Non conta quanto fai, ma quanto amore metti nell'atto che stai compiendo. - Mother Teresa",

"Il segreto del cambiamento è concentrarsi su costruire il nuovo, non combattere l'antico. - Socrate",

"Il successo non è garantito, ma il fallimento è impossibile se non ci si arrende mai. - Unknown",

"La tua mente è come un paracadute, funziona solo se è aperto. - Frank Zappa",

"L'unico modo per fare un grande lavoro è amare quello che fai. - Steve Jobs",

"La gentilezza è una lingua che i sordi possono sentire e i ciechi possono vedere. - Mark Twain",

"La vita è troppo breve per rimpiangere, troppo bella per lamentarsi. - Unknown",

"Sii la migliore versione di te stesso oggi. - Unknown",

"Il successo è un viaggio, non una destinazione. - Ben Sweetland",

"Il modo migliore per predire il futuro è crearlo. - Peter Drucker",

"La felicità non è qualcosa che si posticipa per il futuro; è qualcosa che si progetta per il presente. - Jim Rohn",

"Il miglior modo per prevedere il futuro è crearlo. - Peter Drucker",

"Il coraggio non è l'assenza di paura, ma il trionfo su di essa. - Nelson Mandela",

"Non aspettare l'ispirazione, cerca l'azione. Non aspettare la motivazione, inizia e la motivazione ti troverà. - Unknown",

"Il successo è la somma di piccoli sforzi ripetuti giorno dopo giorno. - Robert Collier",

"La vita è troppo importante per essere presa sul serio. - Oscar Wilde"

"Il successo è come un iceberg. Molte persone vedono solo la punta, ma non conoscono il lavoro svolto sotto la superficie. - Unknown",

"Il tuo tempo è limitato, non sprecarlo vivendo la vita di qualcun altro. - Unknown",

"Il progresso non è mai un risultato accidentale, ma una regola. - Philip J. Romano",

"Il modo migliore per predire il futuro è crearlo. - Peter Drucker",

"La tua attitudine determina la tua direzione. - Unknown",

```

    "Il successo è ottenere ciò che vuoi. La felicità è volere ciò che_
    ↪ottieni. - W.P. Kinsella",
    "Non guardare indietro con rimpianto, guarda avanti con speranza. -_
    ↪Unknown",
    "La vita è ciò che accade quando sei occupato a fare altri progetti. -_
    ↪Allen Sanders",
    "Il modo in cui affronti le sfide determina il tuo successo. - Unknown",
    "Ogni giorno è un'opportunità per un nuovo inizio. - Unknown"
]

#Restituisce una citazione casuale del giorno
return random.choice(citazioni)

#Esempio di utilizzo con la funzione main
def main():
    citazione = genera_citazione_del_giorno()
    print("Buongiorno, la citazione di oggi è:\n")
    print(citazione)
    print("\nAllora non mi resta che augurarti buona giornata e aspettarti per_
    ↪un'altra citazione!")
if __name__ == "__main__":
    main()

```

Buongiorno, la citazione di oggi è:

L'unico modo per fare un grande lavoro è amare quello che fai. - Steve Jobs

Allora non mi resta che augurarti buona giornata e aspettarti per un'altra citazione!

32 LA LETTERATURA COMBINATORIA

Il generatore di personaggi fantasy è basato in questo caso sulla letteratura combinatoria. Con questa funzione, si possono creare personaggi unici e affascinanti combinando elementi letterari in modo creativo. È una forma divertente e innovativa di creare personaggi per le storie fantasy.

```
[7]: print("Buongiorno, questo è un generatore di personaggi fantasy che funziona_
    ↪attraverso la letteratura combinatoria")
print(generatoredipersonaggi())#è lo stesso def del codice del generatore di_
    ↪personaggi fantasy, basta runnare di nuovo quel codice in modo da salvarlo_
    ↪nella memoria e poi da lì il programma riesce a recuperarsi la funzione che_
    ↪avevamo creato in precedenza
```

Buongiorno, questo è un generatore di personaggi fantasy che funziona attraverso la letteratura combinatoria

Per prima cosa scegli il nome con cui vorresti chiamare il personaggio generato:
Matteo

Il personaggio fantasy generato è:

Nome: Matteo

Aspetto fisico: Capelli sottili

Aspetto personale: Hobbyista
Sfondo sociale: Ladro
Motivazione: Essere indipendenti

33 IL GENERATORE DI POST DA INFLUENCER

Questo codice Python implementa un generatore casuale di citazioni, combinando frammenti di citazioni famose in modo creativo. Ogni volta che viene eseguito, il programma seleziona casualmente un numero di frammenti e li rimescola per formare una nuova citazione, offrendo così un'esperienza sempre fresca e ispirante.

```
[10]: import random

# Lista di frammenti di citazioni famose (generate da ChatGPT)
frammenti=[
    "La vita è un'avventura.",
    "Il successo richiede impegno.",
    "Sii creativo.",
    "Non arrenderti mai.",
    "Semplicità ed eleganza.",
    "Ama ciò che fai.",
    "Falllo oggi.",
    "La saggezza del fallimento.",
    "Ogni giorno conta.",
    "Sii audace.",
    "Pensa diversamente.",
    "Credi in te stesso.",
    "La felicità è un viaggio.",
    "Sii il cambiamento che vuoi vedere.",
    "Non avere rimpianti.",
    "Sogna in grande.",
    "Abbraccia il caos.",
    "Lavora sodo, sogna in grande.",
    "Crescita personale.",
    "Sii gentile.",
    "L'arte di ascoltare.",
    "Inseguire i tuoi sogni.",
    "Non limitarti.",
    "Cambia il mondo.",
    "Fai la differenza.",
    "Il potere della positività.",
    "Trova la tua passione.",
    "Fai ciò che ami.",
    "Ogni giorno è un nuovo inizio.",
    "Rischiare è vivere.",
    "Perchè la conoscenza è potere.",
    "Tutto grazie al duro lavoro e alla fatica.",
```

```

    "Tutto grazie al duro lavoro e alla fatica.",  

    "Questo è il segreto del successo!"  

]  
  

# Funzione per creare nuove citazioni rimescolando i frammenti  

def creatoredicitazioni():  

    numeroframmenti=random.randint(5, 7)#Sceglie un numero casuale di frammenti  

    ↪da utilizzare  

    citazionerimescolata=random.sample(frammenti, numeroframmenti)  

    nuovacitazionegenerata=" ".join(citazionerimescolata)  

    return nuovacitazionegenerata  
  

# Genera una nuova citazione  

nuovacitazionegenerata=creatoredicitazioni()  

print("Ciao, benvenuto nel generatore di citazioni")  

print("Tramite questo programma puoi generare delle citazioni in maniera del tutto  

    ↪casuale")  

print("La nuova citazione generata è:")  

print(nuovacitazionegenerata)

```

Ciao, benvenuto nel generatore di citazioni
Tramite questo programma puoi generare delle citazioni in maniera del tutto
casuale
La nuova citazione generata è:
Rischiare è vivere. Tutto grazie al duro lavoro e alla fatica. La felicità è un
viaggio. Pensa diversamente. Credi in te stesso. Ogni giorno è un nuovo inizio.
Non arrenderti mai.

Stesso esercizio ma con frasi diverse e struttura finale leggermente diversa

```

[ ]: import random  
  

# Lista di frammenti di citazioni famose (generate da ChatGPT)  

frammenti=[  

    "Il futuro appartiene a coloro che credono nella bellezza dei propri sogni.",  

    "La vita è fatta di piccoli momenti.",  

    "Non smettere mai di sognare.",  

    "La gentilezza è una lingua che tutti possono capire.",  

    "La vita è troppo breve per essere infelice.",  

    "Ogni giorno è una nuova opportunità per essere una persona migliore.",  

    "Il cambiamento è l'inizio di una nuova avventura.",  

    "L'amore è la forza più potente del mondo.",  

    "La tua volontà è la chiave del tuo successo.",  

    "Il successo inizia con un solo passo.",  

    "La saggezza viene dall'esperienza.",  

    "Il coraggio è fare ciò che è giusto, non ciò che è facile.",  

    "La tua mente è un potente strumento, riempila di pensieri positivi.",  

    "La gratitudine è una medicina per l'anima."
]

```

```

    "Il successo richiede sacrificio.",  

    "La fiducia in se stessi è il primo segreto del successo.",  

    "Il miglior modo per prevedere il futuro è crearlo.",  

    "Non puoi cambiare il passato, ma puoi influenzare il futuro.",  

    "Le persone più felici non hanno tutto, ma fanno il meglio di tutto ciò che  

    ↪hanno.",  

    "Sii il tuo più grande sostenitore.",  

    "Non importa quanto sia difficile, non arrendersi mai.",  

    "La vita è piena di sorprese, abbracciale.",  

    "Sii la migliore versione di te stesso ogni giorno.",  

    "Il successo è il risultato di una mentalità positiva.",  

    "La perseveranza è la chiave del successo.",  

    "Il coraggio è la forza per affrontare le sfide.",  

    "Il futuro appartiene a coloro che credono nella bellezza dei propri sogni.",  

    "Il cambiamento è l'inizio di una nuova avventura.",  

    "La tua mentalità determina la tua realtà.",  

    "Sii il cambiamento che vuoi vedere nel mondo.",  

    "L'unico modo per ottenere ciò che vuoi è credere di meritarlo."
]

```

```

# Funzione per creare nuove citazioni rimescolando i frammenti
def creatoredicitazioni():
    numeroframmenti=random.randint(4, 7)#Sceglie un numero casuale di frammenti
    ↪da utilizzare
    citazionerimescolata=random.sample(frammenti, numeroframmenti)
    nuovacitazionegenerata=" ".join(citazionerimescolata)
    return nuovacitazionegenerata

# Genera una nuova citazione
def main():
    nuovacitazionegenerata = creatoredicitazioni()
    print("Ciao, benvenuto nel generatore di citazioni")
    print("Tramite questo programma puoi generare delle citazioni in maniera del")
    ↪tutto casuale")
    print("La nuova citazione generata è:")
    print(nuovacitazionegenerata)

if __name__ == "__main__":
    main()

```

34 IL GENERATORE DI POESIE

Questo codice in Python è un semplice generatore di poesie casuali. Utilizza liste predefinite di aggettivi, sostantivi e verbi, quindi crea versi combinando casualmente elementi da ciascuna lista. La funzione generatoredipoesia() restituisce una poesia composta da tre versi, ciascuno formato da un aggettivo, un sostantivo e un verbo selezionati casualmente. Il codice può essere descritto come un piccolo esperimento creativo che sfrutta la casualità per generare espressioni poetiche semplici e

imaginative.

```
[20]: import random
#Liste di parole predefinite per la generazione della poesia
aggettivi=["dolce", "sereno", "profondo", "luminoso", "gentile"]
sostantivi=["amore", "mare", "cielo", "vento", "sogno"]
verbi=["danza", "splende", "abbraccia", "canta", "sorride"]

#Genera una poesia casuale
def generatoredi poesia():
    verso1=f"Il {random.choice(aggettivi)} {random.choice(sostantivi)} {random.
    ↪choice(verbi)}."
    verso2=f"Il {random.choice(aggettivi)} {random.choice(sostantivi)} {random.
    ↪choice(verbi)}."
    verso3=f"Nel {random.choice(sostantivi)} {random.choice(verbi)} con {random.
    ↪choice(aggettivi)} {random.choice(sostantivi)}."
    return f"{verso1}\n{verso2}\n{verso3}"
#Stampa la poesia generata
print("Ciao, benvenuto nel generatore di poesie")
print("Tramite questo programma puoi generare delle poesie in maniera del tutto
    ↪casuale")
print("La poesia generata è:")
print(generatoredi poesia())
```

Ciao, benvenuto nel generatore di poesie
Tramite questo programma puoi generare delle poesie in maniera del tutto casuale
La poesia generata è:
Il luminoso sogno canta.
Il profondo sogno abbraccia.
Nel vento sorride con dolce sogno.

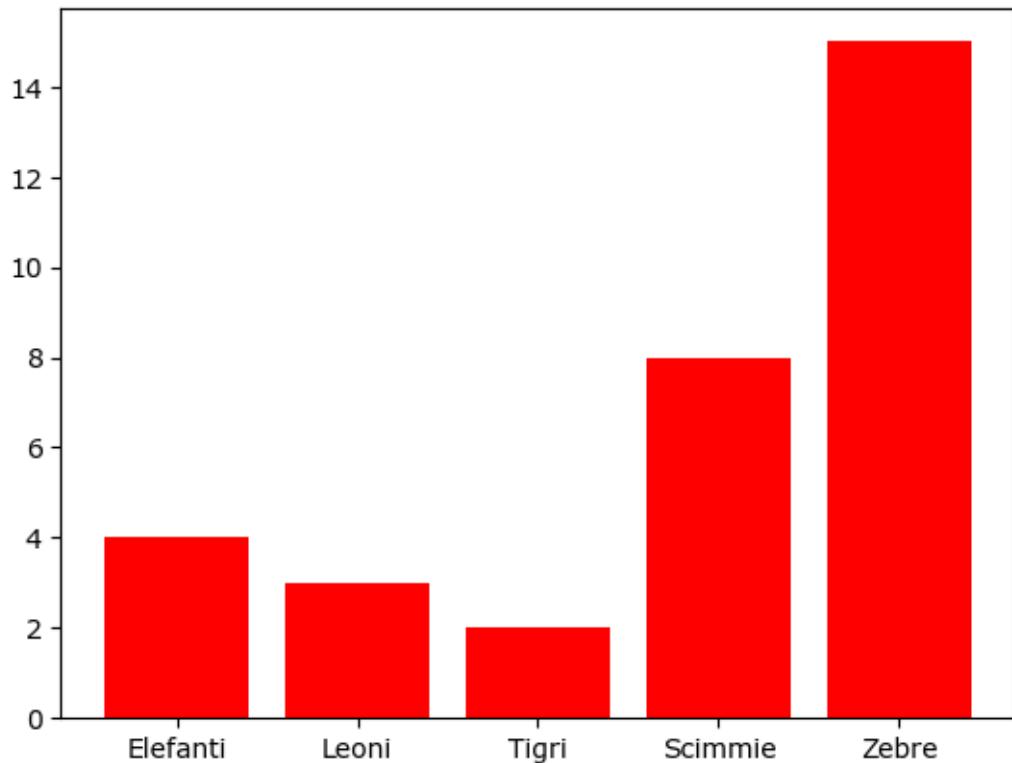
Es. 3 (i grafici) (Maksym)

February 1, 2024

1 I GRAFICI AD ISTOGRAMMA

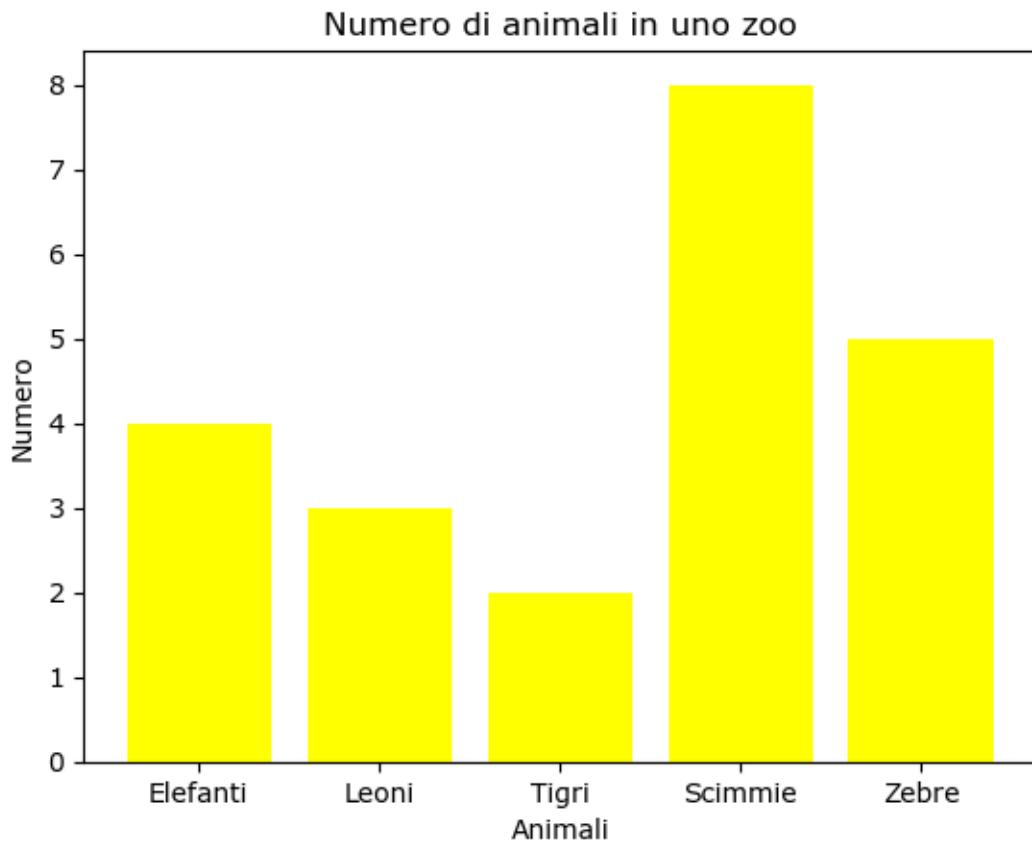
Il codice importa la libreria matplotlib per creare i grafici. Innanzitutto, importiamo la libreria matplotlib con l'alias plt per semplificare, e poi creiamo le liste degli animali e dei numeri. 'plt.bar' è un codice che crea le barre aggiungendo le liste degli animali e dei numeri con il colore rosso.

```
[1]: #Importare la libreria per creare grafici
import matplotlib.pyplot as plt#per semplificare la cosa, si può definire la funzione matplotlib come plt
animalidellozoo=["Elefanti", "Leoni", "Tigri", "Scimmie", "Zebre"]#lista delle specie animali
numeroanimalidellozoo=[4, 3, 2, 8, 15]#lista del numero di esemplari per ciascuna specie
plt.bar(animalidellozoo, numeroanimalidellozoo, color="red") #crea la barra
plt.show() #visualizza il grafico
```



Il codice importa la libreria matplotlib, che serve per creare i grafici, e per semplificare l'uso viene definita come ‘plt’. Il codice crea due liste di animali e numeri e utilizza ‘plt.bar’ per creare le barre con i dati aggiunti. ‘plt.title’ aggiunge un titolo, mentre ‘plt.xlabel’ e ‘plt.ylabel’ aggiungono le etichette sugli assi x e y.

```
[2]: import matplotlib.pyplot as plt #importare la libreria per creare grafici
animalidellozoo=["Elefanti", "Leoni", "Tigri", "Scimmie", "Zebre"] #lista degli animali dello zoo
numeroanimalidellozoo=[4, 3, 2, 8, 5] #numero di ciascun animale nello zoo
plt.bar(animalidellozoo, numeroanimalidellozoo, color="yellow") #creazione di un grafico a barre verticali
plt.title("Numero di animali in uno zoo") #aggiunta del titolo al grafico
plt.xlabel("Animali") #aggiunta dell'etichetta dell'asse x
plt.ylabel("Numero") #aggiunta dell'etichetta dell'asse y
plt.show() #visualizzazione del grafico a barre
```



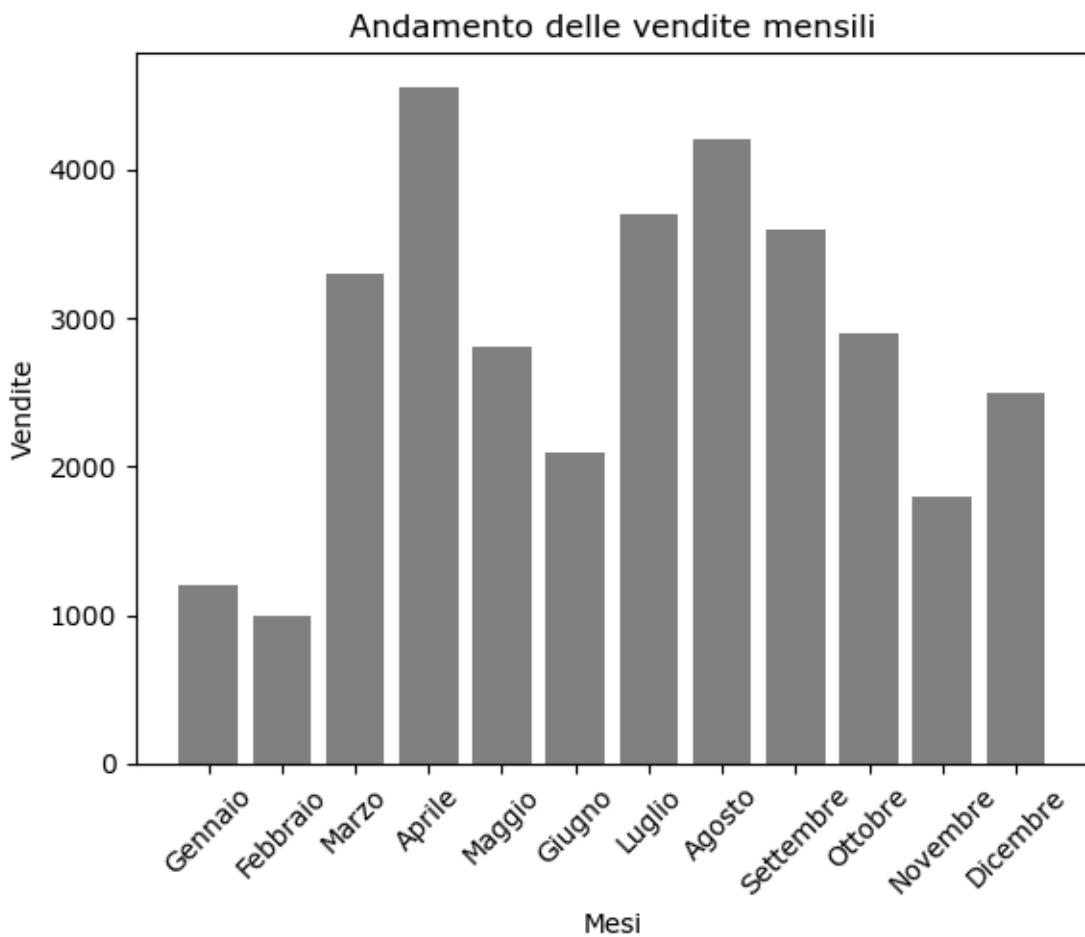
Il codice crea una lista con i mesi e le vendite. Successivamente, aggiunge un titolo e disegna le barre con chiavi e valori utilizzando il colore grigio. Il codice imposta i titoli sull'asse delle 'x' e delle 'y'. Con 'plt.xticks', le parole vengono ruotate di 45 gradi. L'ultima riga visualizza il grafico.

```
[3]: # La lista
venditemensili=
    "Gennaio": 1200,
    "Febbraio": 1000,
    "Marzo": 3300,
    "Aprile": 4555,
    "Maggio": 2800,
    "Giugno": 2100,
    "Luglio": 3700,
    "Agosto": 4200,
    "Settembre": 3600,
    "Ottobre": 2900,
    "Novembre": 1800,
    "Dicembre": 2500
}
plt.title("Andamento delle vendite mensili") #aggiunge il titolo
```

```

plt.bar(venditemensili.keys(),venditemensili.values(),color="grey") #crea la barra
plt.xlabel("Mesi")#titolo asse x
plt.ylabel("Vendite")#titolo asse y
plt.xticks(rotation=45)#imposta l'inclinazione a 45 gradi per i mesi, quindi per i nomi lunghi è più facilmente leggibile
plt.show() #visualizzazione del grafico a barre

```



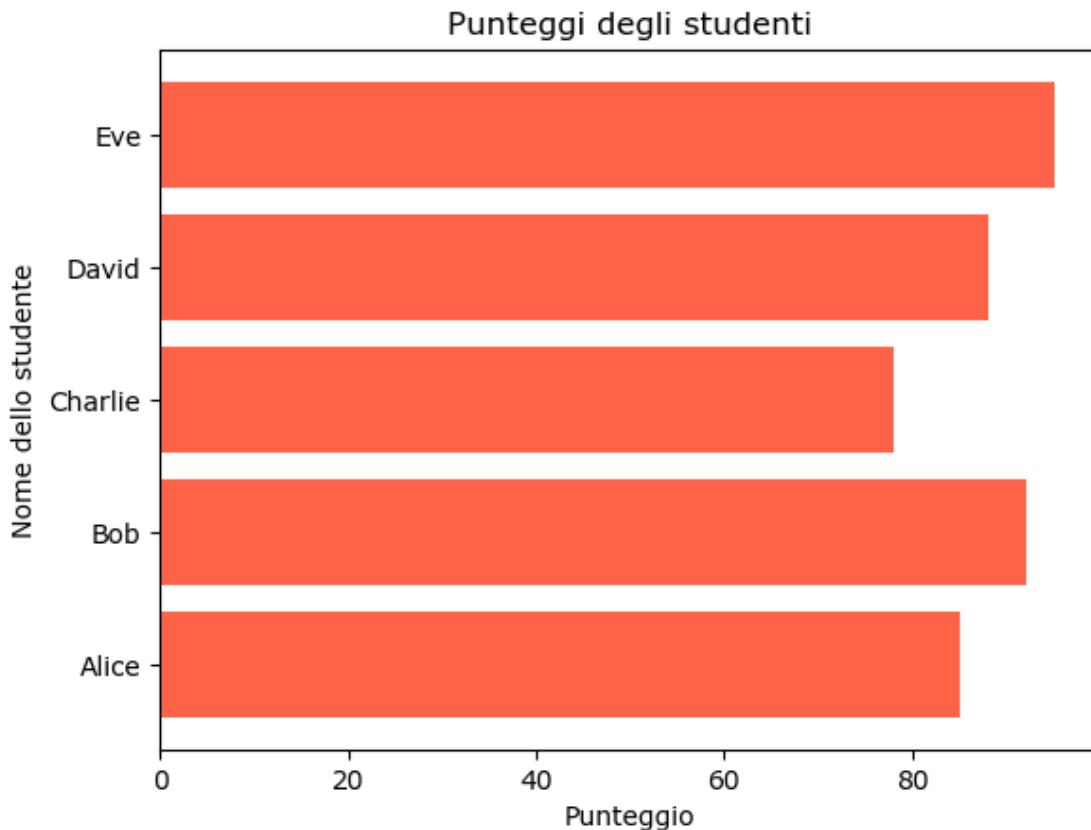
Il codice crea due liste con gli studenti e i punteggi, disegnando le barre in modo orizzontale. Successivamente, aggiunge un titolo e le etichette sugli assi 'x' e 'y', quindi visualizza il grafico.

```

[4]: nomistudenti=['Alice', 'Bob', 'Charlie', 'David', 'Eve'] #La lista dei studenti
punteggi=[85, 92, 78, 88, 95] #La lista dei punteggi
plt.barh(nomistudenti, punteggi, color='tomato')#"barh" a differenza di "bar" inserisce le righe orizzontali
plt.title('Punteggi degli studenti') #Aggiunge il titolo
plt.xlabel('Punteggio') #titolo asse x

```

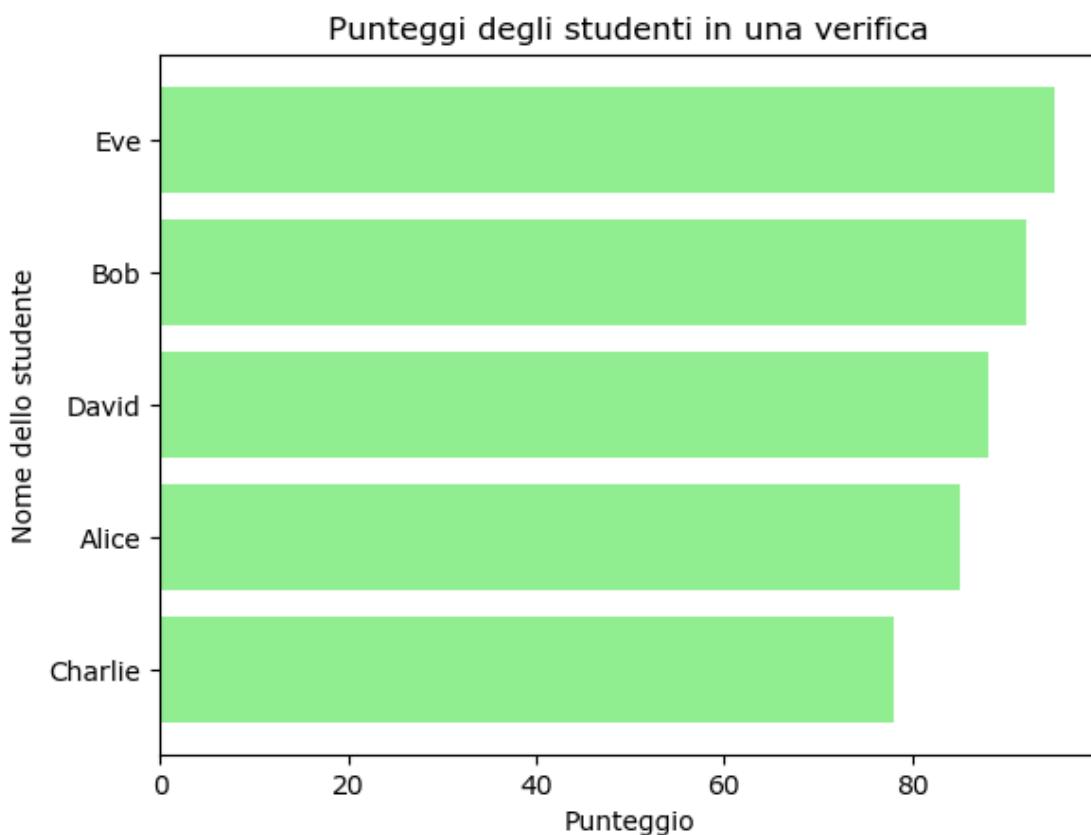
```
plt.ylabel('Nome dello studente') #titolo asse y
plt.show() #visualizzazione del grafico a barre
```



Il codice importa le librerie matplotlib e pandas. Crea le liste con i nomi dei studenti e i punteggi. Successivamente, il codice utilizza queste liste per creare un DataFrame. Ordina il DataFrame in base ai punteggi in ordine crescente, utilizzando ‘inplace=False’ per evitare di sovrapporre i dati con il DataFrame precedente. Infine, il codice crea le barre in modo orizzontale e aggiunge titoli e titoli per gli assi ‘x’ e ‘y’.

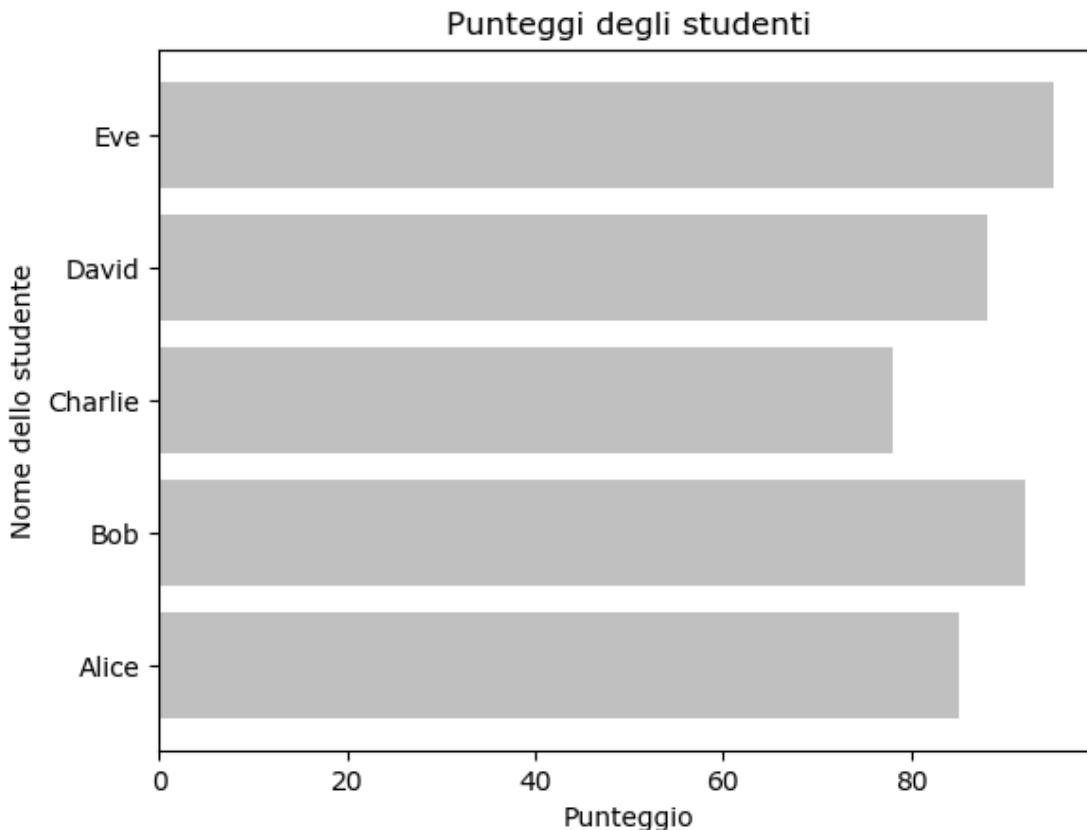
```
[5]: #importa librerie matplotlib e pandas
import matplotlib.pyplot as plt #per semplificare la cosa, si può definire la funzione matplotlib come plt
import pandas as pd #per semplificare la cosa, si può definire la funzione pandas come pd
nomistudenti=['Alice', 'Bob', 'Charlie', 'David', 'Eve'] #La lista dei nomi di studenti
punteggi=[85, 92, 78, 88, 95] #La lista dei punteggi
#Creare un DataFrame con nomi e punteggi
data={'Nome dello studente': nomistudenti, 'Punteggio': punteggi}
df=pd.DataFrame(data)
```

```
#Ordinare il DataFrame per punteggio in ordine crescente
df1=df.sort_values(by='Punteggio', inplace=False) #inplace=False vuol dire che non sovrapporrà i dati con il dataframe precedente
plt.barh(df1['Nome dello studente'], df1['Punteggio'], color='lightgreen') # "barh" a differenza di "bar" inserisce le righe orizzontali
plt.title('Punteggi degli studenti in una verifica') #Aggiunge il titolo
plt.xlabel('Punteggio') #Aggiunge il titolo nelle asse x
plt.ylabel('Nome dello studente')
plt.show() #visualizzazione del grafico a barre
```



Il codice crea le barre in modo orizzontale che richiama il DataFrame se lo si ha runnato e definito e aggiunge titolo e titoli per gli assi 'x' e 'y'.

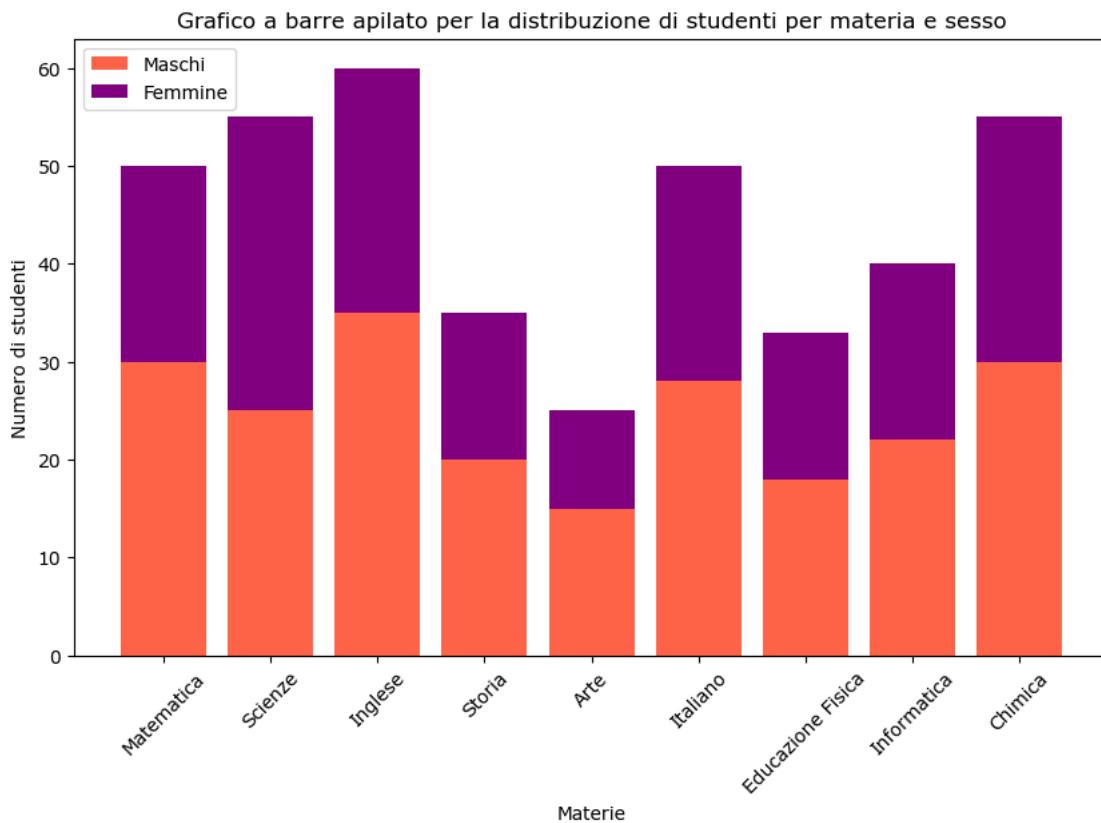
```
[6]: plt.barh(df['Nome dello studente'], df['Punteggio'], color='silver') #si può richiamare il dataframe se lo si ha già runnato prima e definito
plt.title('Punteggi degli studenti') #Aggiunge il titolo
plt.xlabel('Punteggio') #Aggiunge il titolo nella asse x
plt.ylabel('Nome dello studente') #Aggiunge il titolo nella asse y
plt.show() #visualizzazione del grafico a barre
```



Il codice crea le liste per le materie, i maschi e le femmine. Successivamente, imposta le dimensioni del grafico e crea le barre dove sono rappresentati i maschi e le femmine utilizzando colori predefiniti. Viene aggiunto un titolo al grafico e delle etichette sugli assi x e y. Inoltre, il testo sull'asse x viene ruotato di 45 gradi prima di visualizzare il grafico.

```
[7]: materie=['Matematica', 'Scienze', 'Inglese', 'Storia', 'Arte', 'Italiano',  
         'Educazione Fisica', 'Informatica', 'Chimica']#materie nel grafico a barre  
maschi=[30, 25, 35, 20, 15, 28, 18, 22, 30]#numero di studenti per materia  
femmine=[20, 30, 25, 15, 10, 22, 15, 18, 25]#numero di studentesse per materia  
plt.figure(figsize=(10, 6))#imposta le dimensioni del grafico  
plt.bar(materie, maschi, label='Maschi', color='tomato') #Crea la barra con il  
#colore che definito per Maschi  
plt.bar(materie, femmine, label='Femmine', bottom=maschi, color='purple') #Crea  
#la barra con il colore che definito per Femmine  
plt.title('Grafico a barre apilato per la distribuzione di studenti per materia  
e sesso') #Aggiunge il titolo  
plt.xlabel('Materie') #Aggiunge il titolo nella asse x  
plt.ylabel('Numero di studenti') #Aggiunge il titolo nella y  
plt.legend(loc='upper left') #Crea la legenda sotto a sinistra.  
plt.xticks(rotation=45) #ruota il testo nel 45 gradi.
```

```
plt.show() #visualizzazione del grafico a barre
```



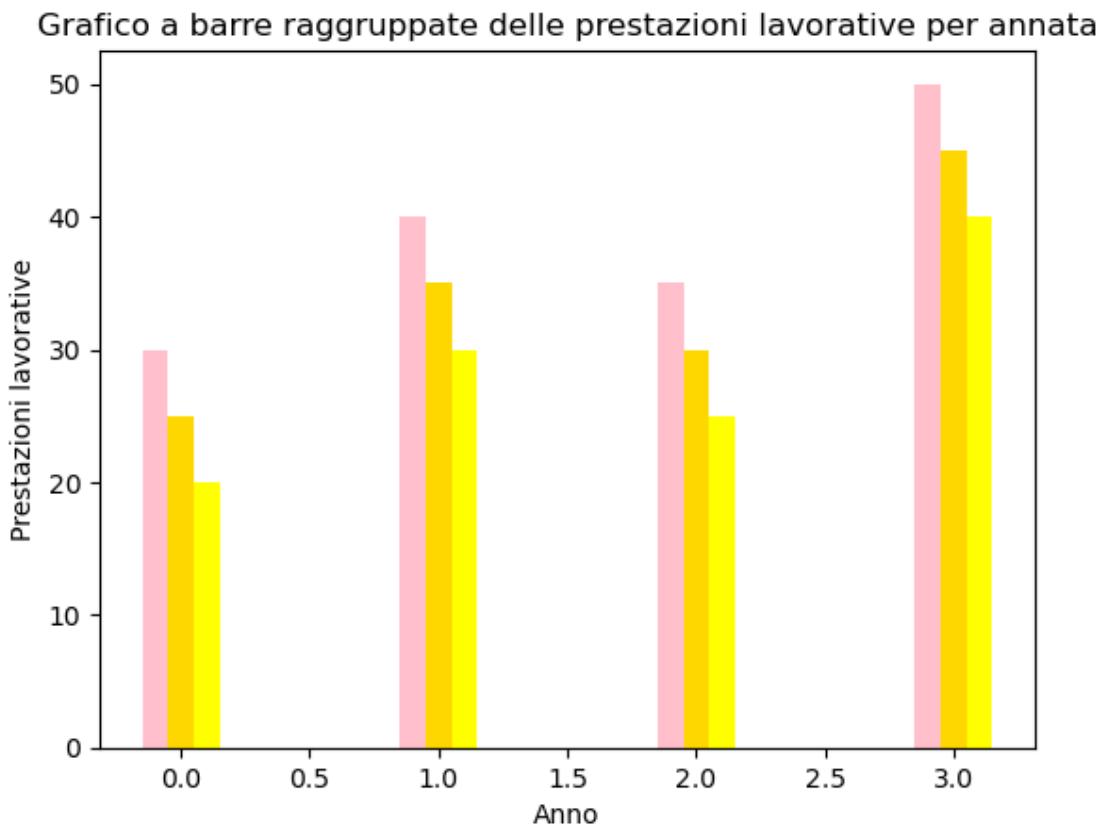
Il codice importa le librerie matplotlib e numpy. Crea le liste e imposta la larghezza delle barre. Successivamente, il codice crea gli indici per posizionare le barre, aggiunge un titolo e le etichette sugli assi x e y.

```
[1]: import matplotlib.pyplot as plt #per semplificare la cosa, si può definire la funzione matplotlib come plt
import numpy as np #per semplificare la cosa, si può definire la funzione numpy come np
annata=['2020', '2021', '2022', '2023'] # La lista
gruppo1=[30, 40, 35, 50] # La lista
gruppo2=[25, 35, 30, 45] # La lista
gruppo3=[20, 30, 25, 40] # La lista
larghezzabarre=0.1 #Imposta la larghezza delle barre
indici=np.arange(len(annata))#larghezza delle barre
#creazione degli indici per posizionare le barre
plt.bar(indici - larghezzabarre, gruppo1, width=larghezzabarre, label='Gruppo 1', color='pink') # Crea le barre per il primo gruppo spostandole a sinistra rispetto all'indice per formare un gruppo
```

```

plt.bar(indici, gruppo2, width=larghezzabarre, label='Gruppo 2', color='gold') # Crea le barre per il secondo gruppo al posto giusto sull'indice
plt.bar(indici + larghezzabarre, gruppo3, width=larghezzabarre, label='Gruppo 3', color='yellow') # Crea le barre per il terzo gruppo spostandole a destra rispetto all'indice per formare un gruppo
plt.title('Grafico a barre raggruppate delle prestazioni lavorative per annata') # Aggiunge il titolo al grafico
plt.xlabel('Anno') # Etichetta l'asse x
plt.ylabel('Prestazioni lavorative') # Etichetta l'asse y
plt.show() # Visualizza il grafico

```



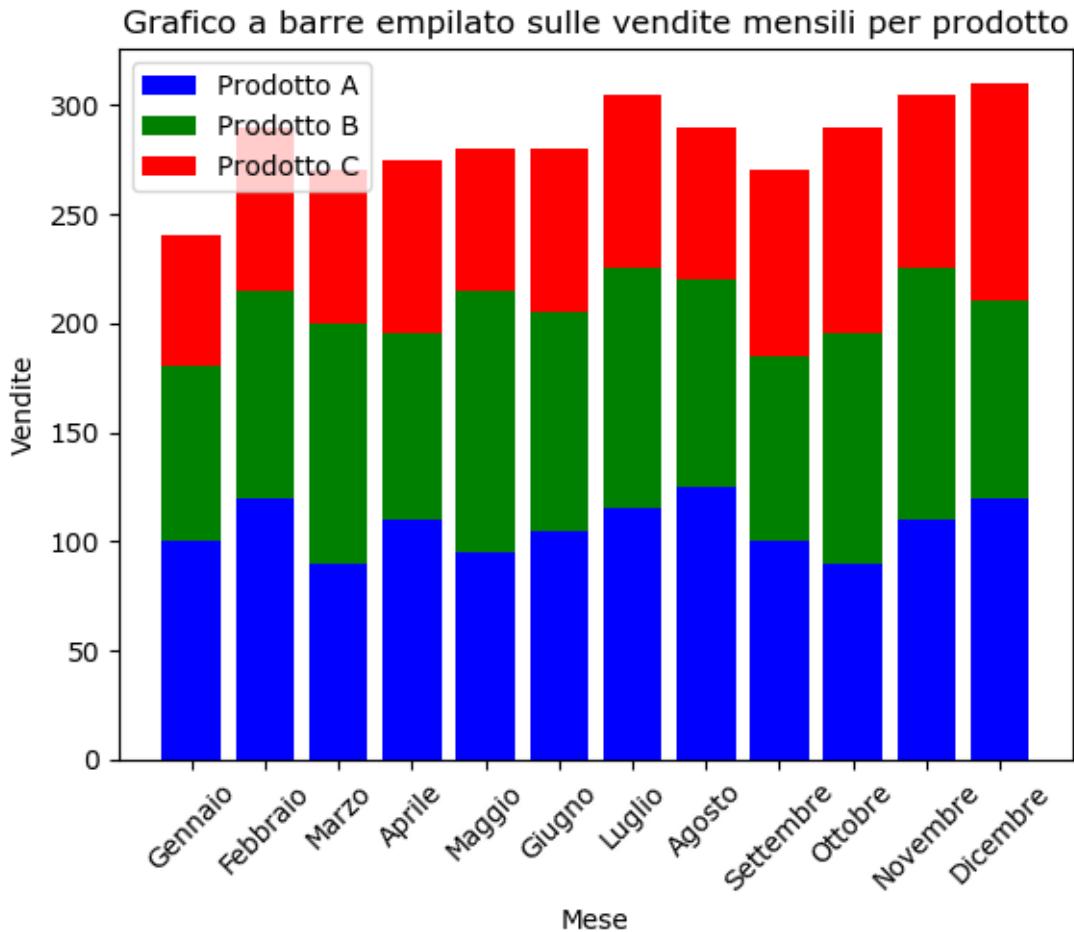
Il codice crea un array che corrisponde alla della lista “annata” stampa un messaggio e stampa la lista.

```
[ ]: indici = np.arange(len(annata)) # Crea un array di indici che corrispondono alla lunghezza della lista 'annata'
print("Le annate sono:") # Stampa un messaggio per indicare che verranno stampate le annate
print(annata) # Stampa la lista delle annate
```

```
[ ]: print("L'array è:")
indici - larghezzabarre
```

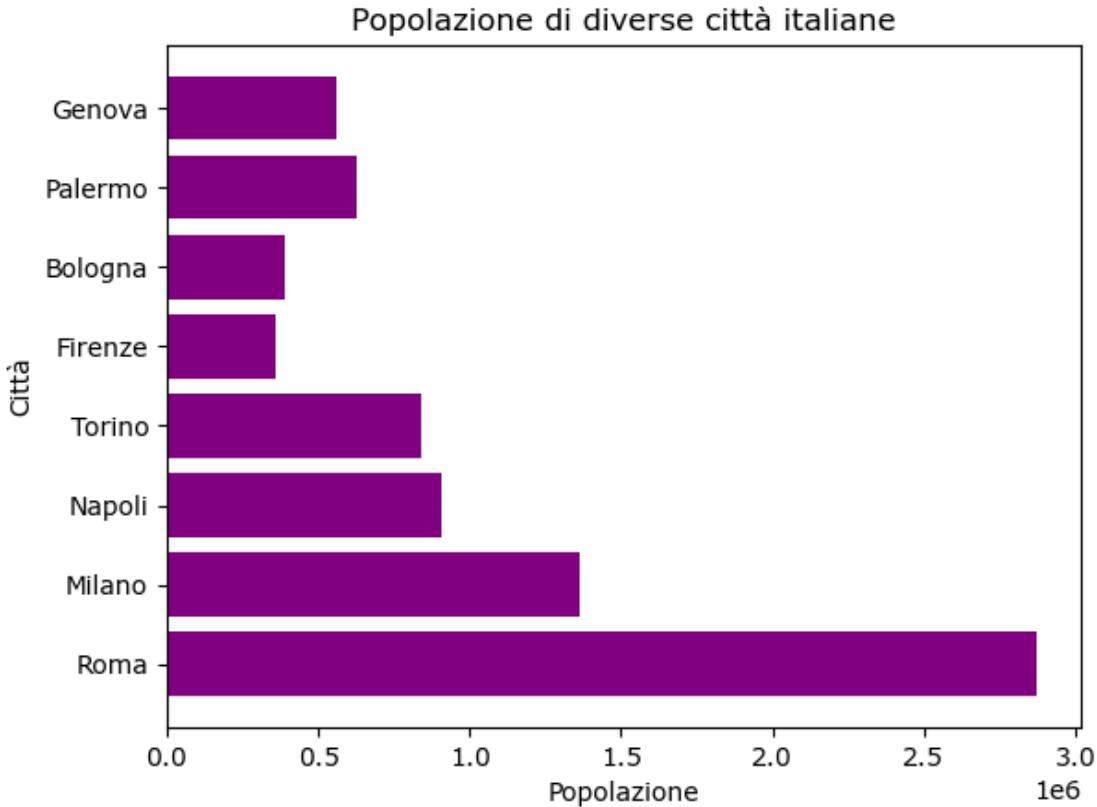
Il codice importa le librerie matplotlib e numpy. Crea le liste, il codice crea gli indici per posizionare le barre, aggiunge una legenda con un titolo e le etichette sugli assi x e y.

```
[1]: import matplotlib.pyplot as plt #per semplificare la cosa, si può definire la
      ↪funzione matplotlib come plt
import numpy as np #per semplificare la cosa, si può definire la funzione numpy
      ↪come np
mesi=['Gennaio', 'Febbraio', 'Marzo', 'Aprile', 'Maggio', 'Giugno', 'Luglio', ↪
      ↪'Agosto', 'Settembre', 'Ottobre', 'Novembre', 'Dicembre'] #Crea la lista
venditeprodottoA=[100, 120, 90, 110, 95, 105, 115, 125, 100, 90, 110, 120] #Crea
      ↪la lista
venditeprodottoB=[80, 95, 110, 85, 120, 100, 110, 95, 85, 105, 115, 90] #Crea la
      ↪lista
venditeprodottoC=[60, 75, 70, 80, 65, 75, 80, 70, 85, 95, 80, 100] #Crea la lista
#creazione un grafico a barre empilato
plt.bar(mesi, venditeprodottoA, label='Prodotto A', color='blue')
plt.bar(mesi, venditeprodottoB, label='Prodotto B', color='green', ↪
      ↪bottom=venditeprodottoA)
plt.bar(mesi, venditeprodottoC, label='Prodotto C', color='red', bottom=np.
      ↪array(venditeprodottoA) + np.array(venditeprodottoB))
plt.title('Grafico a barre empilato sulle vendite mensili per prodotto') #Titolo
plt.xlabel('Mese') #Etichetta asse x
plt.ylabel('Vendite') #Etichetta asse y
plt.legend(loc='upper left') #La legenda
plt.xticks(rotation=45) #ruota il testo nel 45 gradi
plt.show() #visualizzazione del grafico a barre
```



Il codice importa le librerie matplotlib e numpy. Crea le liste, crea le barre in moso orizzontale con colore rosa, aggiunge un titolo e le etichette sugli assi x e y.

```
[2]: import matplotlib.pyplot as plt #per semplificare la cosa, si può definire la funzione matplotlib come plt
import numpy as np #per semplificare la cosa, si può definire la funzione numpy come np
città=['Roma', 'Milano', 'Napoli', 'Torino', 'Firenze', 'Bologna', 'Palermo', 'Genova']
popolazione = [2874605, 1363411, 909422, 837610, 360364, 388001, 627576, 557887]#dati ufficiali cioè controllati su internet
plt.bart(città, popolazione, color='purple') # "bart" a differenza di "bar" inserisce le righe orizzontali
plt.title('Popolazione di diverse città italiane') #Titolo
plt.xlabel('Popolazione') #Etichetta asse x
plt.ylabel('Città') #Etichetta asse y
plt.show() #visualizzazione del grafico a barre
```



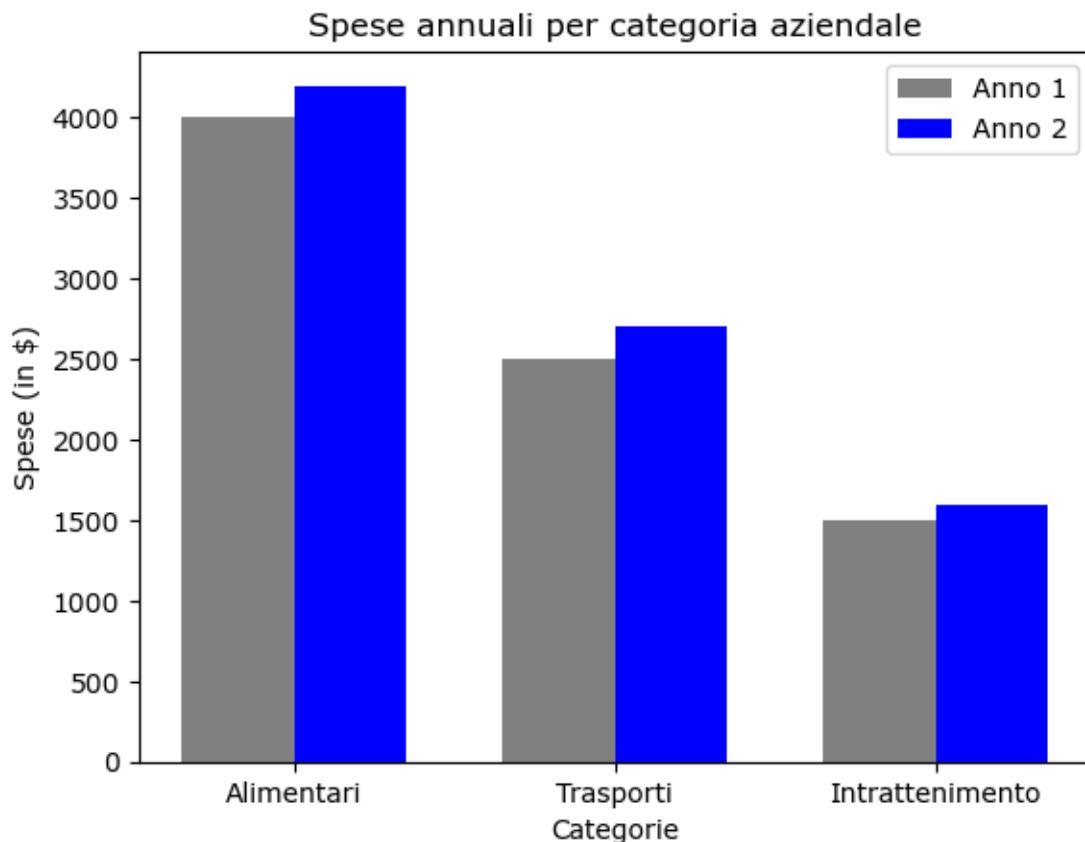
Il codice importa le librerie matplotlib e numpy. Crea le liste e imposta la larghezza delle barre. Successivamente, il codice crea gli indici per posizionare le barre, aggiunge un titolo e le etichette sugli assi x e y.

```
[3]: import matplotlib.pyplot as plt #per semplificare la cosa, si può definire la funzione matplotlib come plt
import numpy as np #per semplificare la cosa, si può definire la funzione numpy come np
categorie=['Alimentari', 'Trasporti', 'Intrattenimento'] #Crea la lista
speseanno1=[4000, 2500, 1500] #Crea la lista
speseanno2=[4200, 2700, 1600] #Crea la lista
larghezzabarre=0.35 #Imposta larghezza delle barre
indici=np.arange(len(categorie)) #Crea un array di indici che corrispondono alla lunghezza della lista 'Categorie'
#creazione un grafico a barre empilato
plt.bar(indici - larghezzabarre/2, speseanno1, width=larghezzabarre, label='Anno 1', color='grey')
plt.bar(indici + larghezzabarre/2, speseanno2, width=larghezzabarre, label='Anno 2', color='blue')
plt.title('Spese annuali per categoria aziendale') #Titolo
plt.xlabel('Categorie') #Etichetta asse x
```

```

plt.ylabel('Spese (in $)') #Etichetta asse y
plt.xticks(indici, categorie) #Imposta le etichette sull'asse x usando gli indici come posizioni e le categorie come etichette
plt.legend(loc='upper right') #Crea la legenda
plt.show() #visualizzazione del grafico a barre

```



2 I GRAFICI A DISPERSIONE DI PUNTI

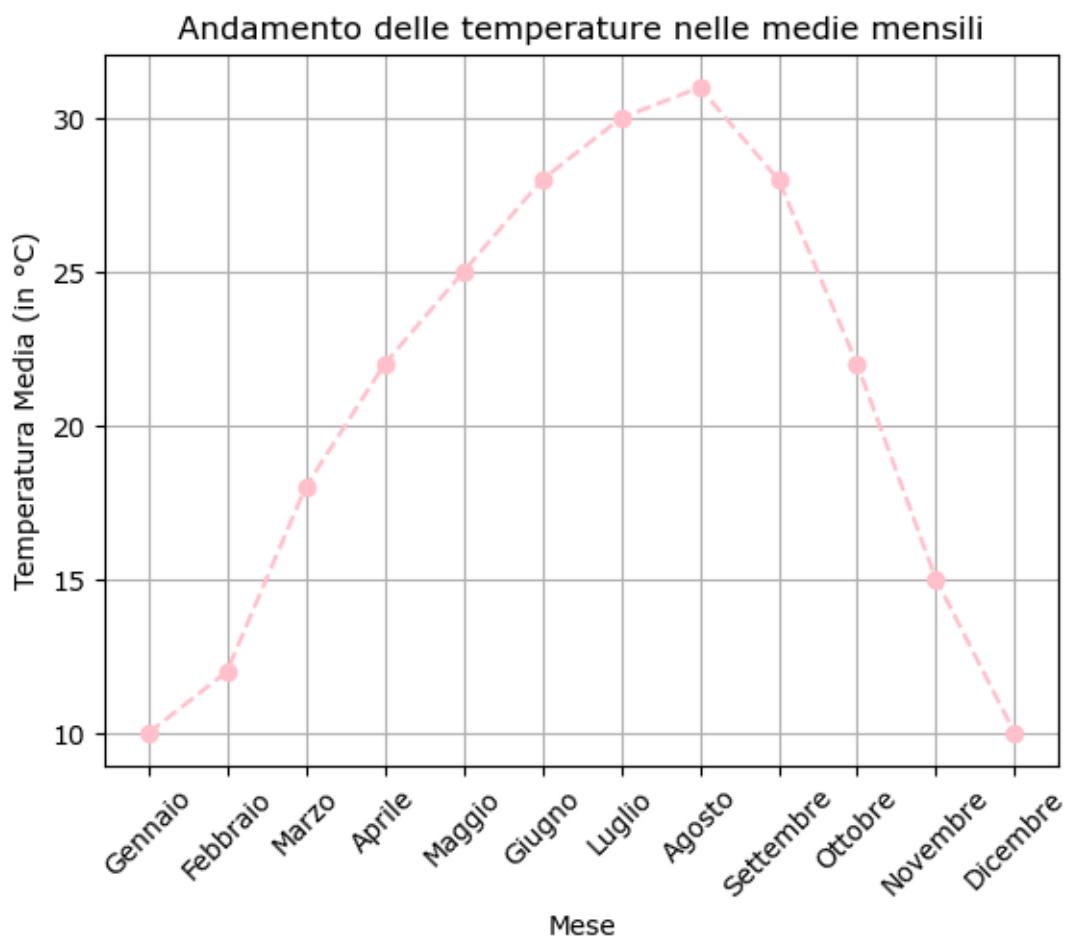
Il codice importa libreria matplotlib e per semplificare si definisce come plt. Il codice crea le liste e usa “plt.plot” serve per creare il grafico, plot serve per creare grafico a dispersione di punti. Il codice il titolo e etichette nelle asse “x” e “y”. Il codice si attiva le griglie e imposta l’inclinazione a 45 gradi.

```
[9]: import matplotlib.pyplot as plt#importare la libreria per creare grafici
mese=["Gennaio", "Febbraio", "Marzo", "Aprile", "Maggio", "Giugno", "Luglio", "Agosto", "Settembre", "Ottobre", "Novembre", "Dicembre"]#lista dei mesi
temperaturamedia=mesi=[10, 12, 18, 22, 25, 28, 30, 31, 28, 22, 15, 10]#lista delle temperature medie mensili
```

```

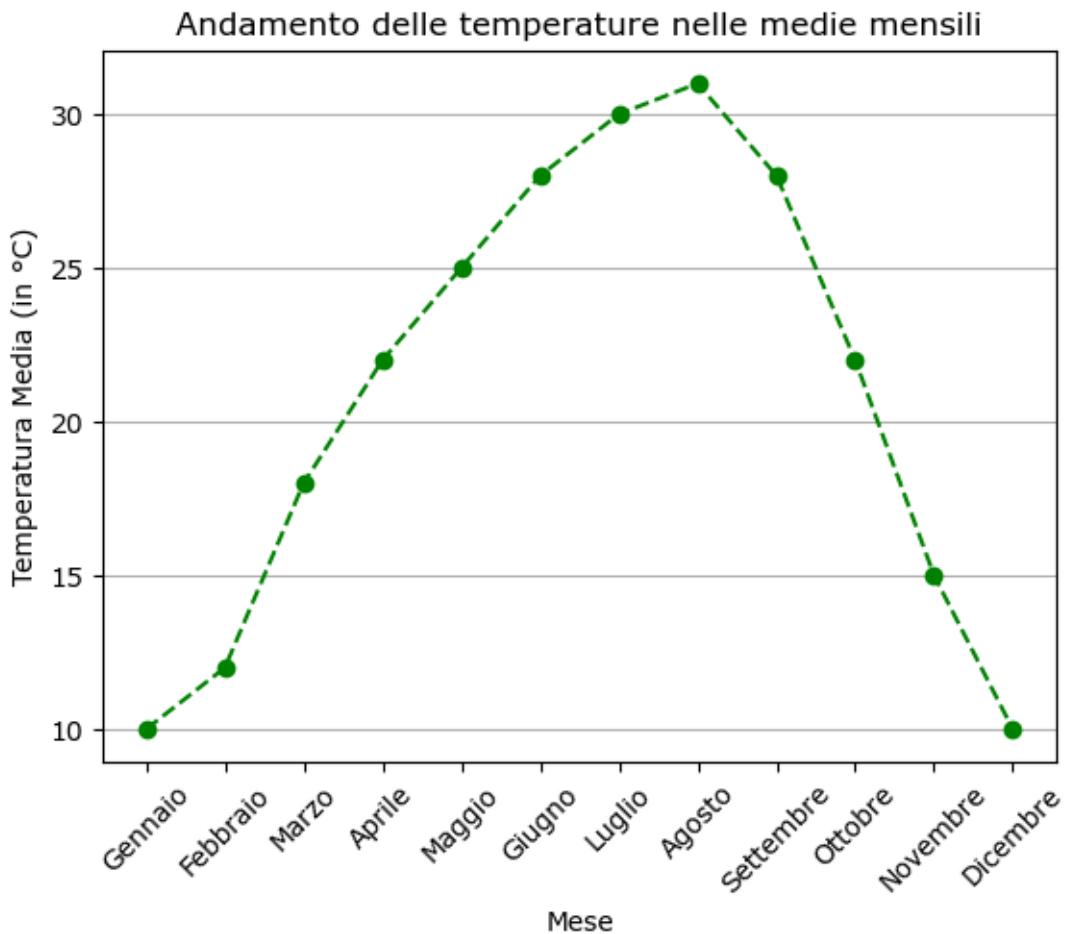
plt.plot(mese, temperaturamediadeimesi, marker="o", linestyle="--",
         color="pink")#creazione del grafico, si usa plot perchè in questo caso è un
         #grafico a dispersione di punti
plt.title("Andamento delle temperature nelle medie mensili")#aggiunta del titolo
         #al grafico
plt.xlabel("Mese")#etichetta dell'asse x
plt.ylabel("Temperatura Media (in °C)")#etichetta dell'asse y
plt.grid(True)#attivazione delle griglie (False disattive, true attive)
plt.xticks(rotation=45)#imposta l'inclinazione a 45 gradi per i mesi, quindi per
         #i nomi lunghi è più facilmente leggibile
plt.show()#visualizzazione del grafico

```



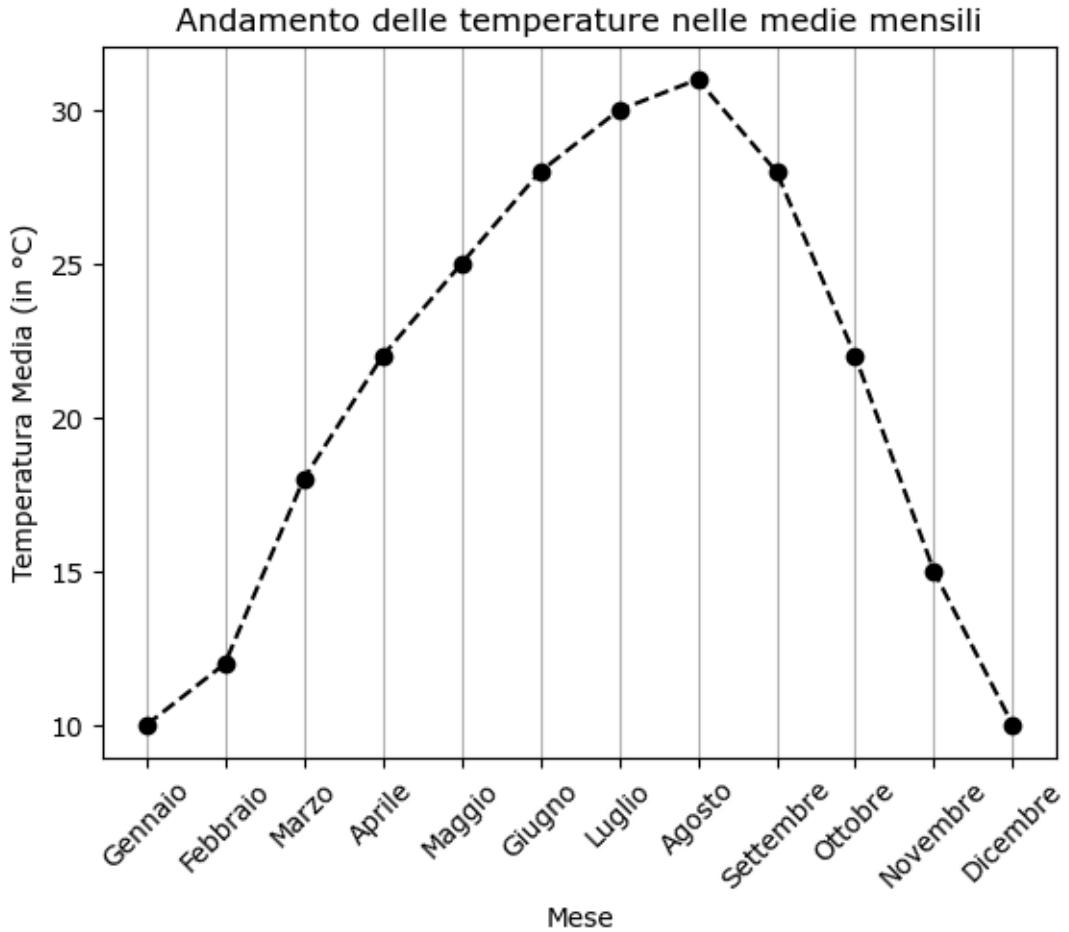
Il codice importa libreria matplotlib e per semplificare si definisce come plt. Il codice crea le liste e usa “plt.plot” serve per creare il grafico, plot serve per creare grafico a dispersione di punti. Il codice il titolo e etichette nelle asse “x” e “y”. Il codice si attiva le griglie e imposta l'inclinazione a 45 gradi.

```
[10]: import matplotlib.pyplot as plt #importare la libreria per creare grafici
mese=["Gennaio", "Febbraio", "Marzo", "Aprile", "Maggio", "Giugno", "Luglio", "Agosto", "Settembre", "Ottobre", "Novembre", "Dicembre"]
temperaturamediadeimesi=[10, 12, 18, 22, 25, 28, 30, 31, 28, 22, 15, 10]
plt.plot(mese, temperaturamediadeimesi, marker="o", linestyle="--", color="green") #creazione del grafico, si usa plot perchè in questo caso è un grafico a dispersione di punti
plt.title("Andamento delle temperature nelle medie mensili")#con "title" si mette il titolo del grafico
plt.xlabel("Mese")#etichetta dell'asse x
plt.ylabel("Temperatura Media (in °C)") #etichetta dell'asse y
plt.grid(True, axis="y")#con l'aggiunta di axis le righe della piano selezionato (in questo caso y) saranno le uniche visibili
plt.xticks(rotation=45)#imposta l'inclinazione a 45 gradi per i mesi, quindi per i nomi lunghi è più facilmente leggibile
plt.show() #visualizzazione del grafico
```



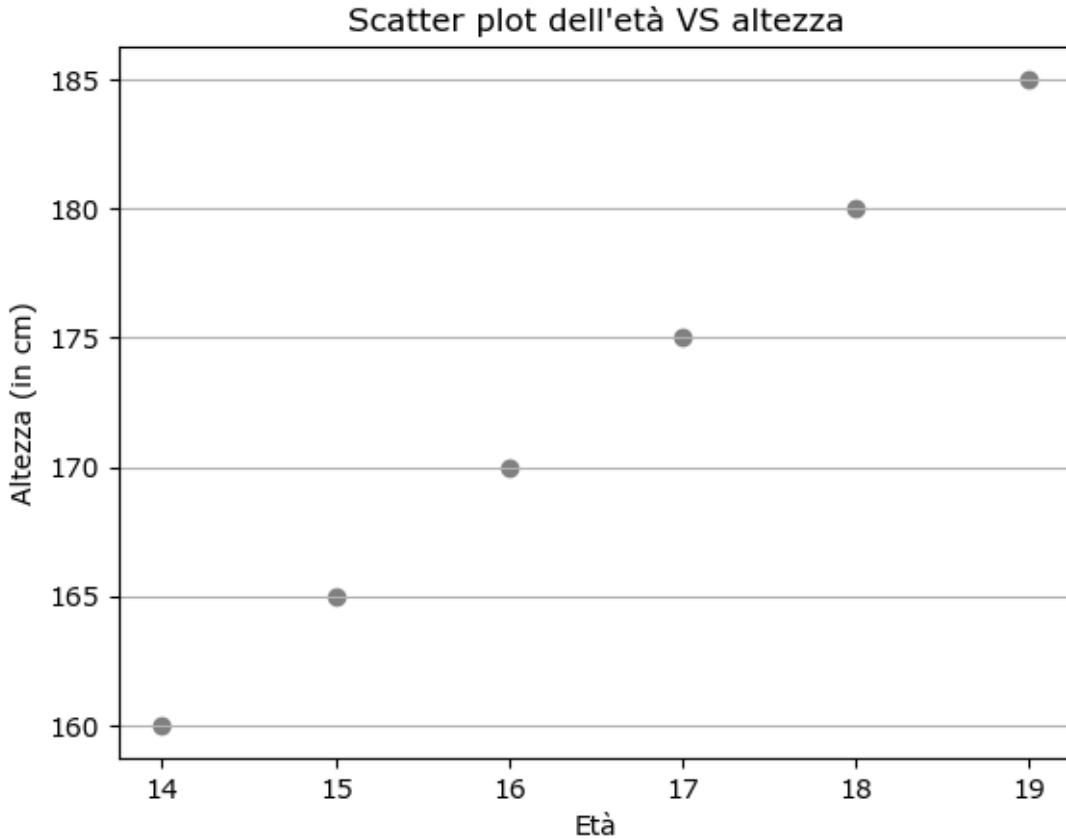
Il codice importa libreria matplotlib e per semplificare si definisce come plt. Il codice crea le liste e usa “plt.plot” serve per creare il grafico, plot serve per creare grafico a dispersione di punti. Il codice il titolo e etichette nelle asse “x” e “y”. Il codice si attiva le griglie e imposta l'inclinazione a 45 gradi.

```
[11]: import matplotlib.pyplot as plt #importare la libreria per creare grafici
mese=["Gennaio", "Febbraio", "Marzo", "Aprile", "Maggio", "Giugno", "Luglio", "Agosto", "Settembre", "Ottobre", "Novembre", "Dicembre"] #Crea la lista
temperaturamediadeimesi=[10, 12, 18, 22, 25, 28, 30, 31, 28, 22, 15, 10] #Crea la lista
plt.plot(mese, temperaturamediadeimesi, marker="o", linestyle="--", color="black") #creazione del grafico, si usa plot perchè in questo caso è un grafico a dispersione di punti
plt.title("Andamento delle temperature nelle medie mensili") #Titolo
plt.xlabel("Mese") #Etichetta asse x
plt.ylabel("Temperatura Media (in °C)") #Etichetta asse y
plt.grid(True, axis="x")#con l'aggiunta di axis le righe della piano selezionato (in questo caso x) saranno le uniche visibili
plt.xticks(rotation=45) #imposta l'inclinazione a 45 gradi per i mesi, quindi per i nomi lunghi è più facilmente leggibile
plt.show() #visualizzazione del grafico
```



Il codice crea le liste, crea uno scatter che ha la forma di un punto e colore grigio. Il codice aggiunge titolo e etichette con le asse “x” e “y”. Il codice crea griglie nella asse y.

```
[12]: età=[14, 15, 16, 17, 18, 19] #Crea la lista
altezza=[160, 165, 170, 175, 180, 185] #Crea la lista
plt.scatter(età, altezza, color='grey', marker='o') #Crea uno scatter plot
    ↪utilizzando le età come coordinata x, altezza come coordinata y, con punti
    ↪color grigio e forma circolare
plt.title("Scatter plot dell'età VS altezza") #Titolo
plt.xlabel("Età") #Etichetta asse x
plt.ylabel("Altezza (in cm)") #Etichetta asse y
plt.grid(True, axis='y') #Crea la griglia nella asse y
plt.show() #visualizzazione del grafico
```



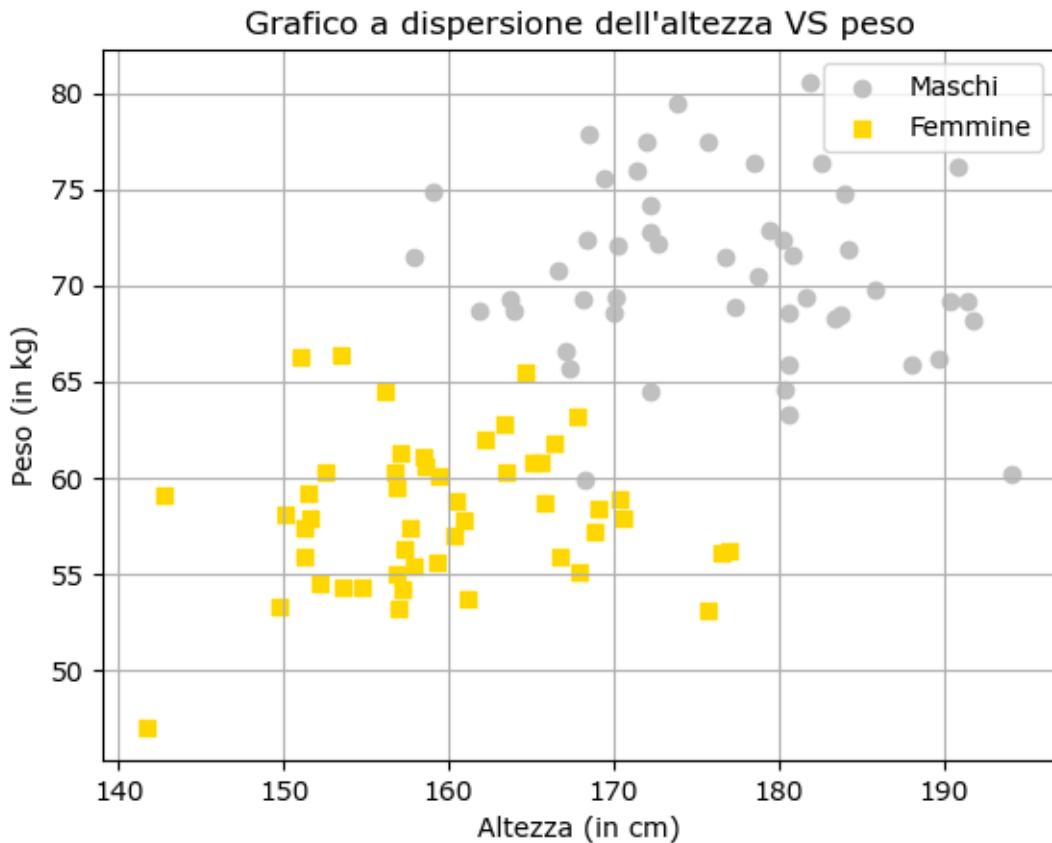
Il codice importa librerie matplotlib e numpy. Il codice sceglie in modo random l'altezza e peso dei maschi e fa stessa cosa per i femmine. Il codice crea il grafico a dispersione per i maschi e femmine. Aggiunge titolo e etichette delle asse "x" e "y". Il codice fa la legenda sopra destra e attiva la griglia.

```
[13]: import matplotlib.pyplot as plt #importare la libreria per creare grafici
import numpy as np #importare la libreria per creare grafici
# Dati di esempio
altezzamaschi=np.random.normal(175, 10, 50)#altezza dei maschi
pesomaschi=np.random.normal(70, 5, 50)#peso dei maschi
altezzafemmine=np.random.normal(162, 8, 50)#altezza delle femmine
pesofemmine=np.random.normal(58, 4, 50)#peso delle femmine
#creazione il grafico a dispersione per i maschi
plt.scatter(altezzamaschi, pesomaschi, color='silver', label='Maschi', marker='o')
#creazione il grafico a dispersione per le femmine
plt.scatter(altezzafemmine, pesofemmine, color='gold', label='Femmine', marker='s')
plt.title("Grafico a dispersione dell'altezza VS peso") #Titolo
plt.xlabel('Altezza (in cm)') #Etichetta asse x
plt.ylabel('Peso (in kg)') #Etichetta asse y
```

```

plt.legend(loc='upper right') #crea la legenda in questo è "right" ma può anche essere "left"
plt.grid(True) #Attiva la griglia
plt.show() #visualizzazione del grafico

```



Il codice crea le liste e crea i grafici a dispersioni con diversi stile a ogni gruppo. Il codice aggiunge titolo e etichette degli asse "x" e "y", mostra la legenda in alto sinistra e attiva la griglia.

```

# Definizione delle liste per gli anni e le prestazioni di ogni gruppo
annata = ['2020', '2021', '2022', '2023', '2024']
gruppo1 = [90, 85, 88, 91, 89]
gruppo2 = [78, 92, 80, 85, 88]
gruppo3 = [85, 79, 91, 94, 92]
gruppo4 = [77, 83, 79, 81, 80]
gruppo5 = [95, 89, 93, 87, 90]

# Grafico a linee per le prestazioni di ogni gruppo per annata
plt.plot(annata, gruppo1, marker='o', label='Gruppo 1', linestyle='-', color='tomato')

```

```

plt.plot(annata, gruppo2, marker='s', label='Gruppo 2', linestyle='--',  

         color='deepskyblue')
plt.plot(annata, gruppo3, marker='^', label='Gruppo 3', linestyle='-.',  

         color='khaki')
plt.plot(annata, gruppo4, marker='v', label='Gruppo 4', linestyle='-.',  

         color='salmon')
plt.plot(annata, gruppo5, marker='x', label='Gruppo 5', linestyle='-.',  

         color='plum')

# Titolo e etichette degli assi
plt.title('Grafico a linee delle prestazioni lavorative di ogni gruppo per  

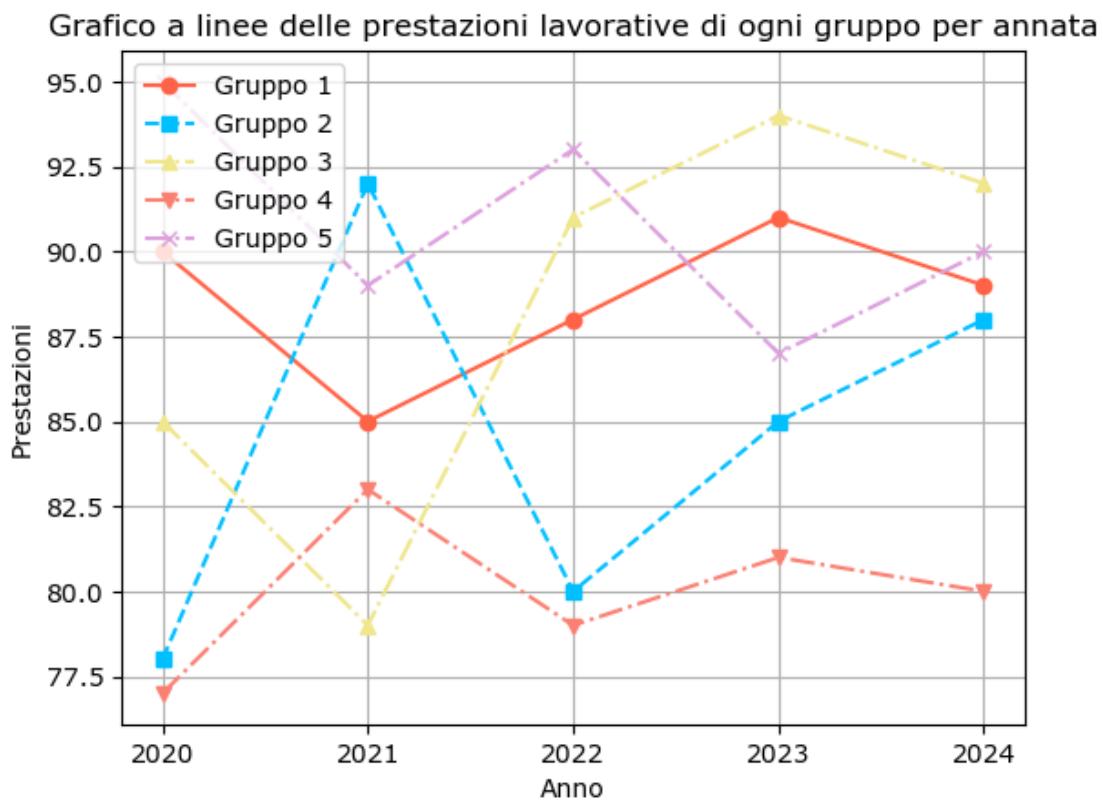
annata')
plt.xlabel('Anno')
plt.ylabel('Prestazioni')

plt.legend(loc='upper left') # Mostra la legenda in alto a sinistra

plt.grid(True) # Abilita la griglia sul grafico

plt.show() # Mostra il grafico

```



3 I GRAFICI A TORTA

Il codice importa la libreria matplotlib e per semplicità e definita come plt. Crea la lista con i colori, mesi e temperatura media. “plt.pie” si usa per fare il grafico a torta. Il codice aggiunge titolo e visualizza grafico.

```
[15]: import matplotlib.pyplot as plt # Importa la libreria matplotlib per la visualizzazione dei grafici

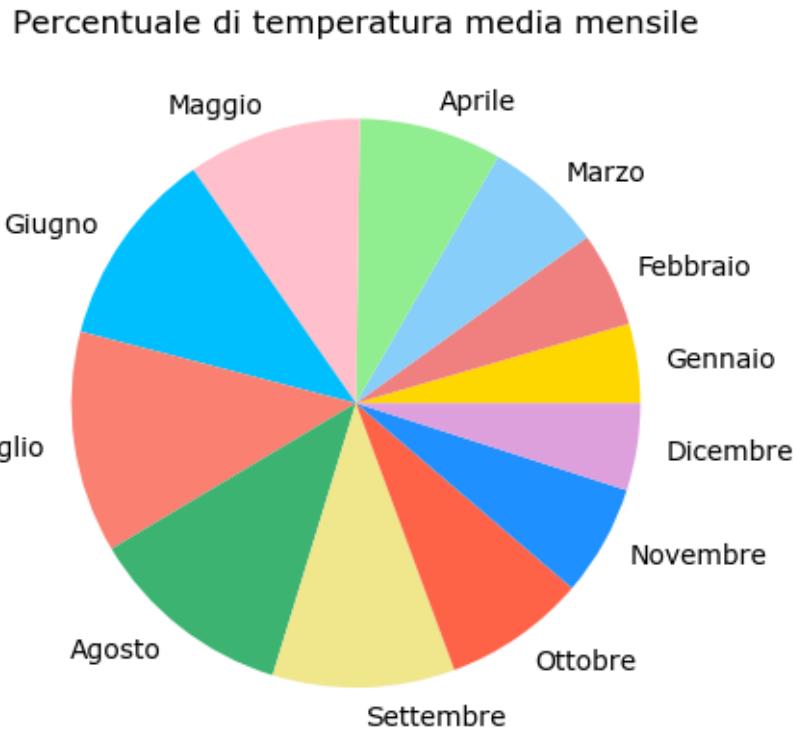
colori = ['gold', 'lightcoral', 'lightskyblue', 'lightgreen', 'pink', 'deepskyblue', 'salmon', 'mediumseagreen', 'khaki', 'tomato', 'dodgerblue', 'plum'] # Definizione dei colori per ciascun mese

mese = ['Gennaio', 'Febbraio', 'Marzo', 'Aprile', 'Maggio', 'Giugno', 'Luglio', 'Agosto', 'Settembre', 'Ottobre', 'Novembre', 'Dicembre'] # Lista dei mesi

temperaturamedia = [10, 12, 15, 18, 22, 25, 28, 26, 23, 18, 14, 11] # Lista delle temperature medie mensili

plt.pie(temperaturamedia, labels=mese, colors=colori) #si usa pie perchè in questo caso è un grafico a torta

plt.title('Percentuale di temperatura media mensile') # Aggiunge un titolo al grafico
plt.show() # Visualizza il grafico
```



Il codice importa la libreria matplotlib e per semplicità e definita come plt. Crea la lista con i colori, mesi e temperatura media viene usata come un label. “plt.pie” si usa per fare il grafico a torta. Il codice aggiunge titolo e visualizza grafico.

```
[16]: import matplotlib.pyplot as plt # Importa la libreria matplotlib per la visualizzazione dei grafici
colori=['deepskyblue', 'salmon', 'mediumseagreen', 'khaki', 'tomato', 'dodgerblue', 'plum'] #Crea la lista con i colori
mese=["Gennaio", "Febbraio", "Marzo", "Aprile", "Maggio", "Giugno", "Luglio", "Agosto", "Settembre", "Ottobre", "Novembre", "Dicembre"]
temperaturamediadeimesi={#al posto di una lista si può usare una labels
    "Gennaio":10,
    "Febbraio":12,
    "Marzo":18,
    "Aprile":22,
    "Maggio":25,
    "Giugno":28,
    "Luglio":30,
    "Agosto":31,
    "Settembre":28,
    "Ottobre":22,
    "Novembre":15,
```

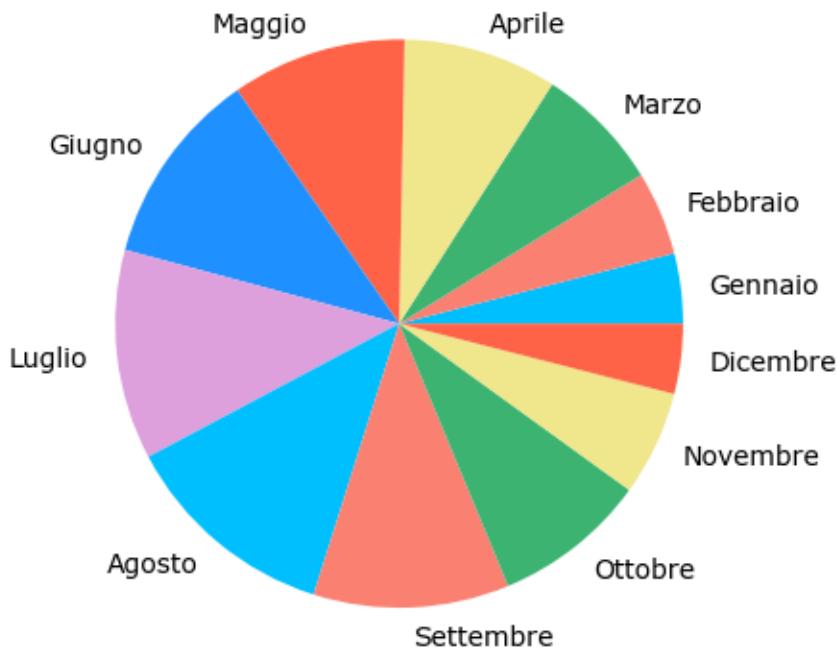
```

    "Dicembre":10
}
plt.pie(temperaturamedia.deimesi.values(), labels=temperaturamedia.deimesi.keys(),  

        colors=colori)#si usa pie perchè in questo caso è un grafico a torta
plt.title('Percentuale di temperatura media mensile') #Titolo
plt.show() #visualizzazione del grafico

```

Percentuale di temperatura media mensile



Il codice importa la libreria matplotlib e per semplicità e definita come plt. Crea la lista con la attività, percentuali e colori. “plt.pie” si usa per fare il grafico a torta. L’argomento “autopct” serve per visualizzare le percentuali sulle fette. Il codice usa “plt.axis” serve per rendere il grafico a forma circolare, aggiunge titolo e visualizza grafico.

```
[17]: import matplotlib.pyplot as plt #Importa la libreria matplotlib per la  

visualizzazione dei grafici  

attività = ['Lavoro', 'Studio', 'Tempo libero'] #La lista di attività  

percentuali = [50, 30, 20] #La lista dei percentuali  

colori=["tomato","purple","silver"] #La lista dei colori  

plt.pie(percentuali, labels=attività, colors=colori, autopct='%1.  

    →1f%%')#"autopct" serve per visualizzare le percentuali sulle fette del grafico  

→a torta con una cifra decimale  

plt.title('Utilizzo del tempo in una giornata tipo') #Titolo  

plt.axis('equal')#rende il grafico a forma circolare, per l'appunto a torta
```

```
plt.show() #visualizzazione del grafico
```



4 LE TABELLE DATI ESEMPIO

Il codice importa librerie matplotlib e pandas, crea le liste con i nomi dei studenti e punteggi. Codice crea un DataFrame con i nomi dei studenti e punteggi, ordina il DataFrame per punteggio in modo crescente e dopo visualizza il DataFrame ordinato.

```
[1]: import matplotlib.pyplot as plt # Importa la libreria matplotlib per la visualizzazione dei grafici
import pandas as pd # Importa la libreria pandas per la manipolazione dei dati

nomistudenti = ['Alice', 'Bob', 'Charlie', 'David', 'Eve'] # Lista dei nomi degli studenti
punteggi = [85, 92, 78, 88, 95] # Lista dei punteggi degli studenti

data={'Nome dello studente': nomistudenti, 'Punteggio': punteggi} #Crea un DataFrame con nomi e punteggi
df=pd.DataFrame(data) #Creazione del DataFrame
```

```

df.sort_values(by='Punteggio', inplace=True) #Ordina il DataFrame per punteggio
    ↵in ordine crescente
df #visualizza il DataFrame ordinato

```

```
[1]: Nome dello studente  Punteggio
      2           Charlie      78
      0            Alice      85
      3           David      88
      1            Bob      92
      4            Eve      95
```

Il codice importa librerie matplotlib e pandas, crea le liste con l'attività, percentuali e colori. Codice crea un dizionario con attività e percentuali, crea un DataFrame e visualizza il DataFrame.

```

[19]: import matplotlib.pyplot as plt # Importa la libreria matplotlib per la
      ↵visualizzazione dei grafici
import pandas as pd # Importa la libreria pandas per la manipolazione dei dati

attività=['Lavoro', 'Studio', 'Tempo libero'] #La lista della attività
percentuali=[50, 30, 20] #Lista dei percentuali
colori=["tomato", "purple", "silver"] #Lista dei colori

data = {'Attività': attività, 'Percentuale': percentuali} # Dizionario con
    ↵attività e percentuali
df = pd.DataFrame(data) # Creazione del DataFrame

df# Visualizza il DataFrame

```

```
[19]: Attività  Percentuale
      0       Lavoro      50
      1       Studio      30
      2  Tempo libero     20
```

Il codice importa librerie matplotlib, pandas e numpy, crea le liste di altezze maschi, peso maschi, altezze femmine, peso femmine e genera i dati in modo random. Crea un DataFrame con i dati generati in modo random e visualizza il DataFrame.

```

[20]: import matplotlib.pyplot as plt # Libreria per la visualizzazione dei grafici
import pandas as pd # Libreria per la manipolazione dei dati
import numpy as np # Libreria numerica per la generazione di dati casuali

altezzamaschi = np.random.normal(175, 10, 50) # Genera 50 valori casuali di
    ↵altezza per i maschi con una media di 175 e deviazione standard di 10
pesomaschi = np.random.normal(70, 5, 50) # Genera 50 valori casuali di peso per i
    ↵maschi con una media di 70 e deviazione standard di 5
altezzafemmine = np.random.normal(162, 8, 50) # Genera 50 valori casuali di
    ↵altezza per le femmine con una media di 162 e deviazione standard di 8

```

```

pesofemmine = np.random.normal(58, 4, 50) # Genera 50 valori casuali di peso per le femmine con una media di 58 e deviazione standard di 4

# Creazione del DataFrame con i dati generati
data = {
    'Altezza maschi': altezzamaschi,
    'Peso maschi': pesomaschi,
    'Altezza femmine': altezzafemmme,
    'Peso femmine': pesofemmme
}
df = pd.DataFrame(data)

df# Visualizzazione del DataFrame

```

	Altezza maschi	Peso maschi	Altezza femmine	Peso femmine
0	179.895355	73.963125	147.018740	51.658344
1	149.234158	75.898661	169.411208	57.962355
2	166.003170	71.405693	157.876062	50.931068
3	171.161443	67.430738	165.956469	52.132664
4	186.408533	65.474049	164.328451	58.047442
5	161.294001	71.068993	152.340138	64.831567
6	187.477734	74.470278	164.035158	61.264310
7	164.016373	70.388611	161.879161	54.180362
8	177.003075	76.635182	159.829225	58.006649
9	181.847934	69.139870	150.132511	58.722132
10	179.165333	65.451794	157.332157	62.967715
11	176.560716	74.269759	161.788424	55.564562
12	165.533504	64.387867	146.741539	58.466281
13	171.135913	63.076959	149.523064	59.106902
14	164.958775	70.449228	166.650440	49.638679
15	181.179053	62.207797	162.972424	59.578701
16	169.208303	79.030138	156.336619	55.876670
17	177.389749	68.871661	172.721423	52.049142
18	167.434088	68.809417	173.041294	59.178438
19	186.077539	71.108472	157.610716	54.613401
20	182.032415	69.938412	168.534666	63.929626
21	190.320113	73.684660	172.447012	51.324119
22	161.990400	74.691392	165.631244	63.689130
23	169.995719	71.847244	159.674249	54.922859
24	159.885023	74.286290	153.029203	56.268316
25	172.064206	70.038041	172.113713	53.366437
26	156.866578	73.612384	158.321993	57.760589
27	170.033356	72.180329	159.381467	65.075837
28	183.429907	63.112686	151.305082	52.550582
29	166.286794	66.285056	153.678976	59.286240
30	163.381342	66.815484	163.073041	60.935934
31	160.078626	66.307460	164.330788	58.259888

32	150.728969	70.117206	171.245676	58.540990
33	166.256364	74.409671	167.059119	51.143821
34	169.880338	70.642900	156.718177	62.157130
35	181.156848	64.820418	158.419378	60.713074
36	163.929348	62.622575	168.470178	64.705295
37	176.082610	67.549719	141.942565	52.142466
38	163.110201	73.390380	170.175281	62.287836
39	178.933943	73.325386	162.824494	56.043555
40	170.950858	67.313469	161.630784	61.360781
41	184.860693	77.147300	162.328225	55.321305
42	183.614716	64.707280	176.339634	57.846557
43	180.000022	61.791613	152.624448	50.488110
44	164.212602	65.195130	155.205527	62.774463
45	180.217245	76.278824	171.041980	69.003169
46	189.308077	60.706863	159.824235	59.150489
47	165.075237	63.776200	162.770393	53.576755
48	151.452458	60.295684	151.971509	55.171644
49	190.561336	64.751759	157.949726	57.829039

Es. 4 (i dataset e i missing values) (Roberto)

February 1, 2024

1 I DATASET E I MISSING VALUES

Il codice analizza un dataset per individuare i valori mancanti e stampa le colonne che contengono dati mancanti insieme al totale di tali dati.

```
[ ]: import pandas as pd

# Dataset con dati mancanti rappresentati da None o NaN
dataset = [
    {"Età": 25, "Punteggio": 90, "Ammesso": 1},
    {"Età": None, "Punteggio": 85, "Ammesso": 0},
    {"Età": 28, "Punteggio": None, "Ammesso": 1},
    {"Età": None, "Punteggio": 75, "Ammesso": 1},
    {"Età": 23, "Punteggio": None, "Ammesso": None},
    {"Età": 23, "Punteggio": 77, "Ammesso": None},
]

# Creazione di un DataFrame pandas dal dataset
df = pd.DataFrame(dataset)

# Conteggio totale dei dati mancanti
totaledatimancanti = df.isnull().sum().sum() # Totale dei valori mancanti in
→tutto il DataFrame

# Determinazione delle colonne con dati mancanti
colonnedatimancanti = df.isnull().any(axis=0) # True se almeno un valore nella
→colonna è mancante

# Stampare le colonne con dati mancanti e il totale dei dati mancanti
print("Colonne con dati mancanti:")
print(colonnedatimancanti)
print(f"Totale dei dati mancanti: {totaledatimancanti}")
```

Il codice analizza un dataset per individuare i valori mancanti, mostrando quali colonne contengono dati mancanti e quante righe hanno almeno un dato mancante.

```
[2]: import pandas as pd

# Definizione del dataset con dati mancanti rappresentati da None o NaN
dataset = [
    {"Età": 25, "Punteggio": 90, "Ammesso": 1},
    {"Età": None, "Punteggio": 85, "Ammesso": 0},
    {"Età": 28, "Punteggio": None, "Ammesso": 1},
    {"Età": None, "Punteggio": 75, "Ammesso": 1},
    {"Età": 23, "Punteggio": None, "Ammesso": None},
    {"Età": 23, "Punteggio": 77, "Ammesso": None},
]

# Creazione di un DataFrame pandas dal dataset
df = pd.DataFrame(dataset)

# Calcolo del totale delle righe con dati mancanti
totaledatimancanti = df.isnull().any(axis=1).sum() # Calcola il totale delle righe con almeno un dato mancante

# Determinazione delle colonne con dati mancanti
colonnedatimancanti = df.isnull().any(axis=0) # True se almeno un valore nella colonna è mancante (None o NaN)

# Stampa delle colonne con dati mancanti e del totale dei dati mancanti
print("Colonne con dati mancanti:")
print(colonnedatimancanti)
print(f"Totale delle righe con dati mancanti: {totaledatimancanti}")
```

Colonne con dati mancanti:

Età	True
Punteggio	True
Ammesso	True
dtype:	bool

Totale delle righe con dati mancanti: 5

Il codice crea un DataFrame da un dataset con valori mancanti rappresentati come None o NaN, quindi rimuove le righe con valori mancanti e stampa entrambi i DataFrame, mostrando prima il DataFrame originale e poi quello con i valori mancanti rimossi.

```
[3]: import pandas as pd

# Definizione del dataset con dati mancanti rappresentati da None o NaN
dataset = [
    {"Nome": "Alice", "Età": 25, "Punteggio": 90, "Email": "alice@email.com"},
    {"Nome": "Bob", "Età": 22, "Punteggio": None, "Email": None},
    {"Nome": "Charlie", "Età": 28, "Punteggio": 75, "Email": "charlie@email.com"},

]
```

```

# Creazione di un DataFrame pandas dal dataset
df = pd.DataFrame(dataset)

# Rimozione delle righe con dati mancanti
df1 = df.dropna(inplace=False) # inplace=False restituisce un nuovo DataFrame
                                ↪con i valori mancanti rimossi

# Stampa del DataFrame originale e del DataFrame dopo la rimozione dei valori
                                ↪mancanti
print(f"Questo è il primo DataFrame con i valori mancanti:\n{df}\nQuesto invece
      ↪è il secondo DataFrame con i valori mancanti rimossi:\n{df1}\n")

```

Questo è il primo DataFrame con i valori mancanti:

	Nome	Età	Punteggio	Email
0	Alice	25	90.0	alice@email.com
1	Bob	22	NaN	None
2	Charlie	28	75.0	charlie@email.com

Questo invece è il secondo DataFrame con i valori mancanti rimossi:

	Nome	Età	Punteggio	Email
0	Alice	25	90.0	alice@email.com
2	Charlie	28	75.0	charlie@email.com

Il codice prepara e visualizza un DataFrame contenente dati di esempio con possibili valori mancanti rappresentati come np.nan nella colonna “Variable2” e come np.nan nella colonna “Colonne mancanti”.

```

[3]: import pandas as pd
import seaborn as sns
import numpy as np
import matplotlib.pyplot as plt

# Dati di esempio
data = {
    'Variable1': [1, 2, 3, 4, 5],                      # Lista di valori per
    ↪Variable1
    'Variable2': [1, 2, np.nan, 4, np.nan],            # Lista di valori per
    ↪Variable2, con valori mancanti rappresentati come np.nan
    'Colonne mancanti': ['A', 'B', 'A', 'C', np.nan] # Lista di valori per
    ↪Colonne mancanti, con un valore mancante rappresentato come np.nan
}

# Creazione di un DataFrame pandas dai dati
df = pd.DataFrame(data)

# Stampa del DataFrame
print(df)

```

```
[3]: Variable1  Variable2 Colonne mancanti
0          1          1.0          A
1          2          2.0          B
2          3          NaN          A
3          4          4.0          C
4          5          NaN          NaN
```

Il codice crea un DataFrame da un dataset di dizionari, poi rimuove le righe contenenti valori mancanti e stampa entrambi i DataFrame, mostrando i dati originali e quelli con i valori mancanti rimossi.

```
[4]: import pandas as pd

# Creazione di un nuovo DataFrame vuoto
df1 = pd.DataFrame()

# Selezione delle colonne numeriche dal DataFrame originale e riempimento dei
# valori NaN con la media
colonneneconomiche = df.select_dtypes(include=['number']) # Selezione delle
# colonne numeriche
df1[colonneneconomiche.columns] = colonneneconomiche.fillna(colonneneconomiche.mean())
# Riempiendo dei valori NaN con la media

# Selezione delle colonne categoriche dal DataFrame originale e riempimento dei
# valori NaN con la moda
colonnecategoriche = df.select_dtypes(include=['object']) # Selezione delle
# colonne categoriche
df1[colonnecategoriche.columns] = colonnecategoriche.fillna(colonnecategoriche.
mode().iloc[0]) # Riempiendo dei valori NaN con la moda

# Stampa del DataFrame originale e del DataFrame con i valori mancanti sostituiti
print(f"Questo è il DataFrame originale con i valori mancanti: \n{df}\n")
print(f"Questo invece è il DataFrame con i missing values sostituiti:\n{df1}\n")
```

Questo è il DataFrame originale con i valori mancanti:

	Nome	Età	Punteggio	Email
0	Alice	25	90.0	alice@email.com
1	Bob	22	NaN	None
2	Charlie	28	75.0	charlie@email.com

Questo invece è il DataFrame con i missing values sostituiti:

	Età	Punteggio	Nome	Email
0	25	90.0	Alice	alice@email.com
1	22	82.5	Bob	alice@email.com
2	28	75.0	Charlie	charlie@email.com

Il codice crea un DataFrame da un dataset di dizionari, seleziona le colonne categoriche e riempie i

valori mancanti di queste colonne con la moda, quindi stampa sia il DataFrame originale che quello con i valori mancanti sostituiti.

```
[1]: import pandas as pd
import seaborn as sns
import numpy as np
import matplotlib.pyplot as plt

# Dati di esempio
data = {
    'Variable1': [1, 2, 3, 4, 5],
    'Variable2': [1, 2, np.nan, 4, np.nan],
    'Colonne mancanti': ['A', 'B', 'A', 'C', np.nan]
}

# Creazione di un DataFrame pandas dal dataset
df = pd.DataFrame(data)
df1 = pd.DataFrame()

# Seleziona le colonne numeriche e riempি i valori NaN con la media
colonneneumeriche = df.select_dtypes(exclude=['number']) # Selezione delle
    ↵colonne numeriche
# Seleziona le colonne categoriche e riempি i valori NaN con la moda
colonnecategoriche = df.select_dtypes(include=['object']) # Selezione delle
    ↵colonne categoriche
df1[colonnecategoriche.columns] = colonnecategoriche.fillna(colonnecategoriche.
    ↵mode().iloc[0])

# Stampa dei DataFrame originale e modificato
print(f"Questo è il DataFrame originale con i valori mancanti: \n{df}\n")
print(f"Questo invece è il DataFrame con i missing values sostituiti:\n{df1}\n")
```

Questo è il DataFrame originale con i valori mancanti:

	Variable1	Variable2	Colonne mancanti
0	1	1.0	A
1	2	2.0	B
2	3	NaN	A
3	4	4.0	C
4	5	NaN	NaN

Questo invece è il DataFrame con i missing values sostituiti:

	Colonne mancanti
0	A
1	B
2	A
3	C
4	A

Il codice riempie i valori mancanti nelle colonne numeriche di un DataFrame utilizzando la media dei valori numerici presenti. Successivamente, viene stampato un messaggio temporaneo per indicare che il problema è stato affrontato temporaneamente. Infine, i valori mancanti nelle colonne numeriche vengono nuovamente riempiti solo con la media dei valori numerici, escludendo eventuali valori non numerici presenti nelle colonne, e il DataFrame risultante viene stampato.

```
[7]: # Riempি i valori mancanti nelle colonne numeriche con la media
df1[colonneneriche.columns] = colonneneriche.fillna(colonneneriche.mean())

# Stampa un messaggio temporaneo per mostrare come il problema sia stato affrontato temporaneamente
print("Per risolvere temporaneamente il problema ho fatto questo:")

# Riempি i valori mancanti nelle colonne numeriche solo con la media dei valori numerici
df1[colonneneriche.columns] = colonneneriche.fillna(colonneneriche.mean(numeric_only=True))

# Stampa il DataFrame con i valori mancanti corretti
print(df1)
```

Per risolvere temporaneamente il problema ho fatto questo:

```
Colonne mancanti
0          A
1          B
2          A
3          C
4        NaN
```

```
/var/folders/54/_j1x1btd64ng49h3vm6864gr0000gn/T/ipykernel_77630/2610008769.py:2
: FutureWarning: The default value of numeric_only in DataFrame.mean is
deprecated. In a future version, it will default to False. In addition,
specifying 'numeric_only=None' is deprecated. Select only valid columns or
specify the value of numeric_only to silence this warning.
```

```
df1[colonneneriche.columns] =
colonneneriche.fillna(colonneneriche.mean())
```

Il codice genera un DataFrame con dati di esempio, riempie i valori mancanti con la media per le colonne numeriche e con la moda per le colonne categoriche, quindi crea grafici a barre per confrontare le medie delle variabili originali e riempite

2 L'USO DEI GRAFICI PER MOSTRARE I DATAFRAME

```
[8]: import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
```

```

# Generazione di dati di esempio
data = {
    'Variable n°1': [1, 2, 3, 4, 5],
    'Variable n°2': [1, 2, np.nan, 4, np.nan],
    'Colonne mancanti': ['A', 'B', 'A', 'C', np.nan]
}

# Creazione di un DataFrame pandas dal dataset di esempio
df = pd.DataFrame(data)
df1 = pd.DataFrame()

# Selezione delle colonne numeriche e riempimento dei valori NaN con la media
colonnenumeriche = df.select_dtypes(include=['number'])
df1[colonnenumeriche.columns] = colonnenumeriche.fillna(colonnenumeriche.mean())

# Selezione delle colonne categoriche e riempimento dei valori NaN con la moda
colonnecategoriche = df.select_dtypes(exclude=['number'])
df1[colonnecategoriche.columns] = colonnecategoriche.fillna(colonnecategoriche.
    mode().iloc[0])

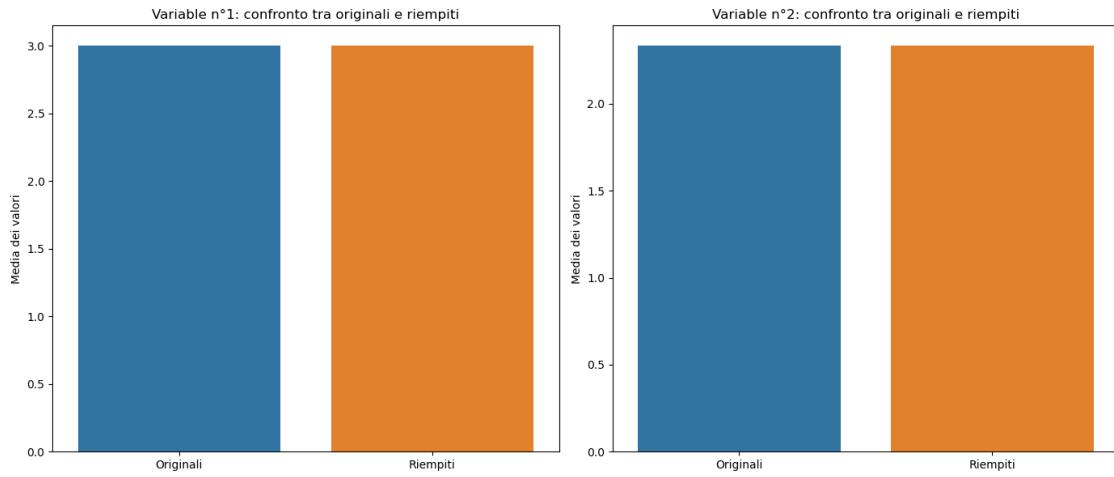
# Creazione di grafici a barre per confrontare le medie delle variabili
→numeriche originali e riempite
plt.figure(figsize=(14, 6))

# Grafico per la Variable n°1
plt.subplot(1, 2, 1)
sns.barplot(x=['Originali', 'Riempiti'], y=[df['Variable n°1'].mean(),
    df1['Variable n°1'].mean()])
plt.title('Variable n°1: confronto tra originali e riempiti')
plt.ylabel('Media dei valori')

# Grafico per la Variable n°2
plt.subplot(1, 2, 2)
sns.barplot(x=['Originali', 'Riempiti'], y=[df['Variable n°2'].mean(),
    df1['Variable n°2'].mean()])
plt.title('Variable n°2: confronto tra originali e riempiti')
plt.ylabel('Media dei valori')

# Mostra i grafici
plt.tight_layout()
plt.show()

```



Crea un DataFrame con dati di esempio, sostituisce i valori mancanti con la media per le variabili numeriche e con il valore più comune per le variabili categoriche, quindi stampa entrambi i DataFrame.

```
[9]: import pandas as pd
import seaborn as sns
import numpy as np
import matplotlib.pyplot as plt

# Dati di esempio
data = {
    'Variable1': [1, 2, 3, 4, 5],
    'Variable2': [1, 2, np.nan, 4, np.nan],
    'Missing Column': ['A', 'B', 'A', 'C', np.nan]
}

# Creazione di un DataFrame
df = pd.DataFrame(data)
df1 = pd.DataFrame()

def gestionemissingvalues(df):
    # Trattamento dei valori mancanti nelle variabili numeriche e categoriche
    colonnenumeriche = df.select_dtypes(include=['number'])
    colonnecategoriche = df.select_dtypes(exclude=['number'])

    # Riempri i valori mancanti nelle variabili numeriche con la media e nelle
    # variabili categoriche con il valore più comune
    df1[colonnenumeriche.columns] = df[colonnenumeriche.columns].mean()
    df1[colonnecategoriche.columns] = df[colonnecategoriche.columns].mode().iloc[0]
```

```

    return df1

def main():
    # Chiama la funzione per gestire i valori mancanti
    df1 = gestionemissingvalues(df)

    # Stampa il DataFrame originale e quello con i valori mancanti sostituiti
    print(f"Questo è il DataFrame originale con i valori mancanti:\n{df}\n")
    print(f"Questo invece è il DataFrame con i valori mancanti sostituiti:
→\n{df1}\n")

if __name__ == "__main__":
    main()

```

Questo è il DataFrame originale con i valori mancanti:

	Variable1	Variable2	Missing	Column
0	1	1.0		A
1	2	2.0		B
2	3	NaN		A
3	4	4.0		C
4	5	NaN		NaN

Questo invece è il DataFrame con i missing values sostituiti:

	Variable1	Variable2	Missing	Column
0	1	1.000000		A
1	2	2.000000		B
2	3	2.333333		A
3	4	4.000000		C
4	5	2.333333		A

Il codice genera dati casuali riguardanti età, genere, punteggio e reddito e utilizza le librerie Seaborn e Plotly per visualizzare vari tipi di grafici statistici. Questi includono un istogramma dell'età, un grafico a barre del conteggio per genere, uno scatter plot di età vs punteggio, e sia un box plot che un violin plot per la distribuzione del reddito per genere. Infine, viene creato uno scatter plot interattivo con Plotly che mostra la relazione tra età, punteggio e reddito.

```
[11]: import pandas as pd # Importa la libreria pandas per la gestione dei dati
import numpy as np # Importa la libreria numpy per la generazione di dati
→casuali
import matplotlib.pyplot as plt # Importa la libreria matplotlib per la
→visualizzazione dei dati
import seaborn as sns # Importa la libreria seaborn per la visualizzazione
→avanzata dei dati
import plotly.express as px # Importa la libreria plotly express per la
→creazione di grafici interattivi
```

```

# Dati puramente casuali a solo scopo scientifico
np.random.seed(42) # Imposta il seme casuale per rendere i risultati
→ riproducibili

# Genera un dizionario di dati casuali contenenti età, genere, punteggio e
→ reddito
data = {
    'Età': np.random.randint(18, 70, size=1000), # Genera età casuali comprese
→ tra 18 e 70 anni
    'Genere': np.random.choice(['Maschio', 'Femmina'], size=1000), # Genera
→ genere casuali (Maschio o Femmina)
    'Punteggio': np.random.uniform(0, 100, size=1000), # Genera punteggi
→ casuali compresi tra 0 e 100
    'Reddito': np.random.normal(50000, 15000, size=1000) # Genera redditi
→ casuali con distribuzione normale
}

# Crea un DataFrame pandas dai dati generati
df = pd.DataFrame(data)

# Istogramma dell'età
plt.figure(figsize=(10, 6)) # Imposta le dimensioni della figura
sns.histplot(df['Età'], kde=True) # Crea un istogramma con una stima della
→ densità kernel
plt.title('Distribuzione dell\'età') # Imposta il titolo del grafico
plt.xlabel('Età') # Etichetta l'asse x
plt.ylabel('Frequenza') # Etichetta l'asse y
plt.show() # Mostra il grafico

# Grafico a barre del genere
plt.figure(figsize=(10, 6)) # Imposta le dimensioni della figura
sns.countplot(x='Genere', data=df) # Crea un grafico a barre del conteggio per
→ genere
plt.title('Conteggio per genere') # Imposta il titolo del grafico
plt.xlabel('Genere') # Etichetta l'asse x
plt.ylabel('Conteggio') # Etichetta l'asse y
plt.show() # Mostra il grafico

# Scatter plot dell'età vs punteggio
plt.figure(figsize=(10, 6)) # Imposta le dimensioni della figura
sns.scatterplot(x='Età', y='Punteggio', data=df) # Crea uno scatter plot di età
→ vs punteggio
plt.title("Relazione tra l'età e il punteggio") # Imposta il titolo del grafico
plt.xlabel('Età') # Etichetta l'asse x
plt.ylabel('Punteggio') # Etichetta l'asse y
plt.show() # Mostra il grafico

```

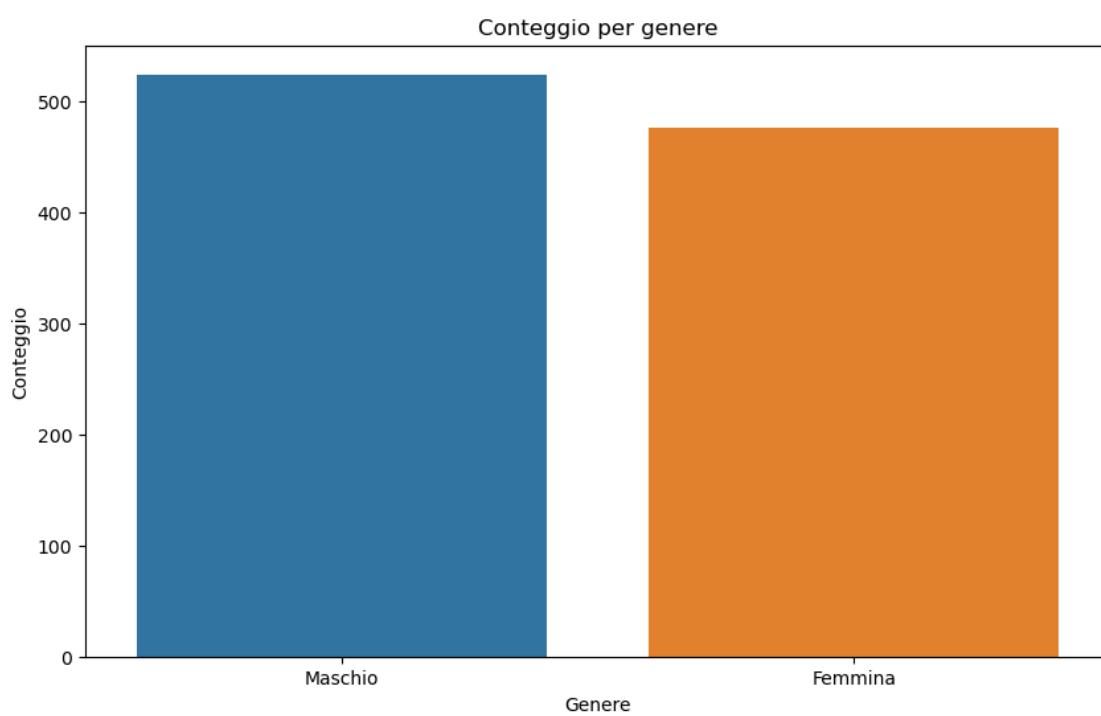
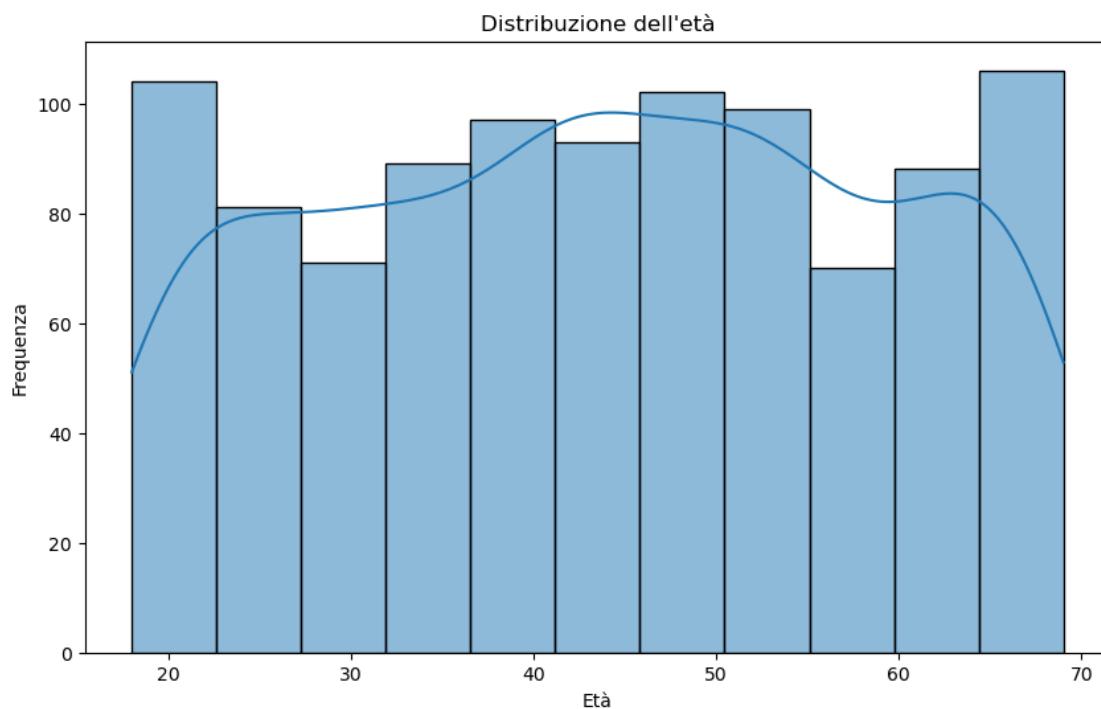
```

# Box plot del reddito per genere
plt.figure(figsize=(10, 6)) # Imposta le dimensioni della figura
sns.boxplot(x='Genere', y='Reddito', data=df) # Crea un box plot del reddito per genere
plt.title('Distribuzione del reddito per genere') # Imposta il titolo del grafico
plt.xlabel('Genere') # Etichetta l'asse x
plt.ylabel('Reddito') # Etichetta l'asse y
plt.show() # Mostra il grafico

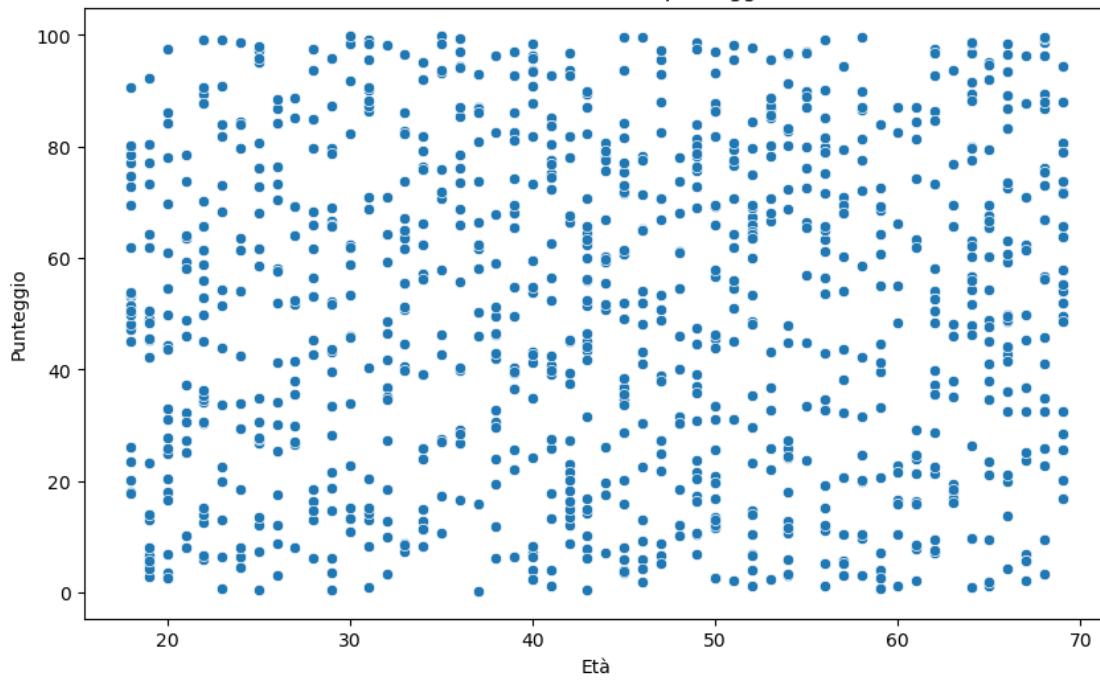
# Violin plot del reddito per genere
plt.figure(figsize=(10, 6)) # Imposta le dimensioni della figura
sns.violinplot(x='Genere', y='Reddito', data=df) # Crea un violin plot del reddito per genere
plt.title('Distribuzione del reddito per genere') # Imposta il titolo del grafico
plt.xlabel('Genere') # Etichetta l'asse x
plt.ylabel('Reddito') # Etichetta l'asse y
plt.show() # Mostra il grafico

# Plotly Scatter Plot dell'età, punteggio e reddito
fig = px.scatter(df, x='Età', y='Punteggio', color='Reddito', hover_data=['Genere']) # Crea uno scatter plot interattivo con plotly express
fig.update_layout(title="Relazione tra l'età, il punteggio e il reddito", xaxis_title='Età', yaxis_title='Punteggio') # Aggiorna il layout del grafico
fig.show() # Mostra il grafico

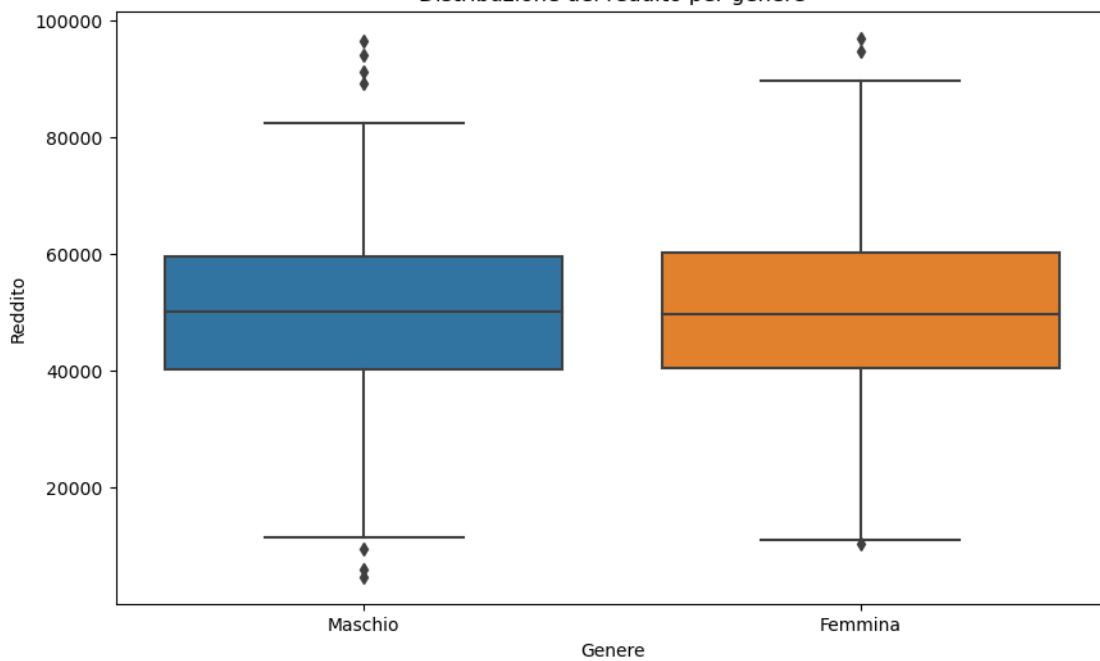
```

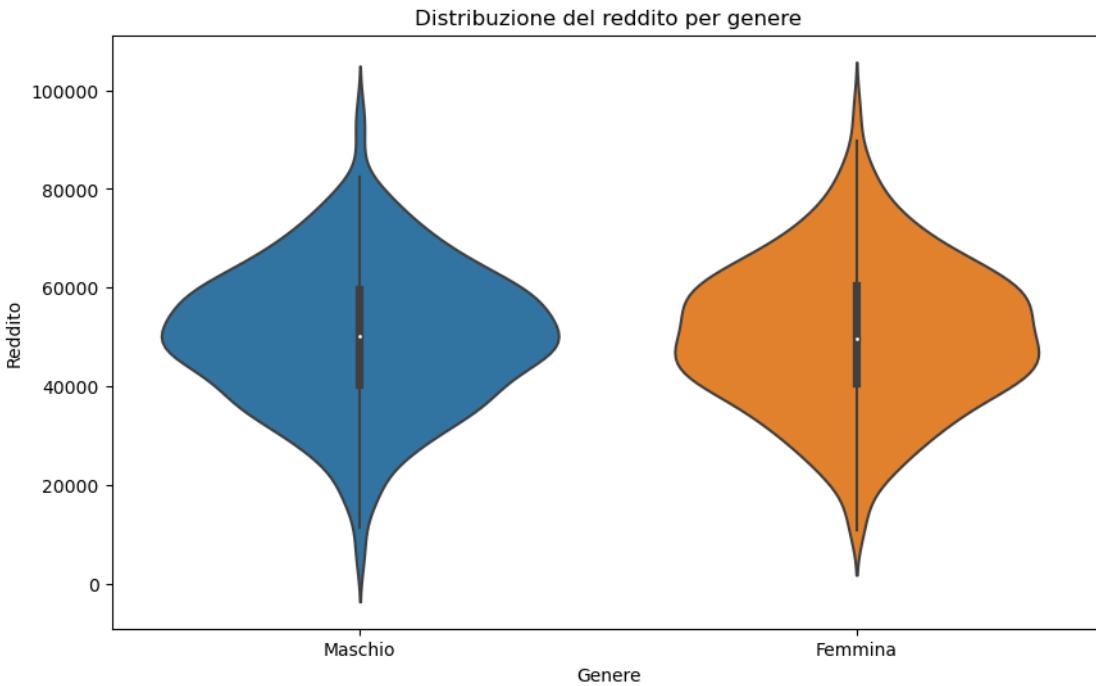


Relazione tra l'età e il punteggio



Distribuzione del reddito per genere





Questo codice genera un DataFrame contenente dati casuali su età, genere, punteggio e reddito, quindi stampa le prime 50 righe del DataFrame e successivamente stampa le righe dall'indice 12 all'indice 52 del DataFrame.

```
[6]: import pandas as pd # Importa la libreria pandas per la gestione dei dati
      import numpy as np # Importa la libreria numpy per la generazione di dati
      ↵casuali

      # Genera dati casuali riguardanti età, genere, punteggio e reddito
      np.random.seed(42) # Imposta il seme casuale per rendere i risultati
      ↵riproducibili
      data = {
          'Età': np.random.randint(18, 70, size=1000), # Genera età casuali comprese
          ↵tra 18 e 70 anni
          'Genere': np.random.choice(['Maschio', 'Femmina'], size=1000), # Genera
          ↵genere casuali (Maschio o Femmina)
          'Punteggio': np.random.uniform(0, 100, size=1000), # Genera punteggi
          ↵casuali compresi tra 0 e 100
          'Reddito': np.random.normal(50000, 15000, size=1000) # Genera redditi
          ↵casuali con distribuzione normale
      }

      df = pd.DataFrame(data) # Crea un DataFrame pandas dai dati generati
```

```

# Stampa le prime 50 righe del DataFrame
print("Ecco il dataframe che inizia da 0 fino a 49:")
print(df.head(50)) # Prende le prime 50 righe del DataFrame e le stampa

# Stampa le righe dal 13° al 52° del DataFrame
print("Ecco il dataframe che inizia da 12 fino a 52:")
print(df.iloc[12:53]) # Stampa le righe dall'indice 12 all'indice 52 del DataFrame

```

	Età	Genere	Punteggio	Reddito
0	56	Maschio	85.120691	52915.764524
1	69	Maschio	49.514653	44702.505608
2	46	Maschio	48.058658	55077.257652
3	32	Femmina	59.240778	45568.978848
4	60	Maschio	82.468097	52526.914644
5	25	Maschio	34.780921	69763.963034
6	38	Maschio	67.801615	34901.861521
7	56	Femmina	56.573196	67098.178417
8	36	Femmina	26.702827	69756.726010
9	40	Maschio	87.862999	48228.972095
10	28	Maschio	79.742602	18172.176515
11	28	Maschio	65.845183	40882.670194
12	41	Maschio	85.058173	69454.918198
13	53	Femmina	86.729420	49656.979066
14	57	Femmina	70.836298	35010.466516
15	41	Femmina	83.701333	42428.376182
16	20	Femmina	69.747146	62609.300398
17	39	Maschio	68.014077	58201.003523
18	19	Maschio	61.861138	46416.018512
19	41	Maschio	75.271664	44497.633823
20	61	Femmina	15.860511	44123.627764
21	47	Maschio	88.087076	36163.847233
22	55	Femmina	87.184353	74230.635342
23	19	Femmina	2.924728	45165.192962
24	38	Maschio	82.581675	68257.377812
25	50	Maschio	12.886987	72819.740779
26	29	Maschio	33.511885	64974.663472
27	39	Femmina	74.350826	43525.695309
28	61	Maschio	16.075990	56055.951355
29	42	Femmina	81.796702	49637.065904
30	66	Maschio	83.213418	36444.472089
31	44	Femmina	50.746773	54865.389226
32	59	Femmina	0.638587	32314.403133
33	45	Maschio	28.703813	67815.190827
34	33	Femmina	61.692692	43030.740541
35	32	Femmina	98.118618	53017.394838
36	64	Maschio	63.181353	54249.318073
37	68	Femmina	25.980358	46116.425394

38	61	Maschio	63.400570	58800.407022
39	69	Femmina	53.998538	42876.445151
40	20	Maschio	77.984540	63069.459462
41	54	Maschio	10.698064	29810.304796
42	68	Femmina	76.102790	51895.693693
43	24	Femmina	54.126658	79083.934986
44	38	Femmina	96.299200	34995.030131
45	26	Femmina	34.187217	39833.825442
46	56	Femmina	63.262189	57708.617745
47	35	Maschio	93.202811	52693.726729
48	21	Femmina	10.250973	55259.451489
49	42	Maschio	93.722849	57337.806952
		Età	Genere	Punteggio Reddito
12	41	Maschio	85.058173	69454.918198
13	53	Femmina	86.729420	49656.979066
14	57	Femmina	70.836298	35010.466516
15	41	Femmina	83.701333	42428.376182
16	20	Femmina	69.747146	62609.300398
17	39	Maschio	68.014077	58201.003523
18	19	Maschio	61.861138	46416.018512
19	41	Maschio	75.271664	44497.633823
20	61	Femmina	15.860511	44123.627764
21	47	Maschio	88.087076	36163.847233
22	55	Femmina	87.184353	74230.635342
23	19	Femmina	2.924728	45165.192962
24	38	Maschio	82.581675	68257.377812
25	50	Maschio	12.886987	72819.740779
26	29	Maschio	33.511885	64974.663472
27	39	Femmina	74.350826	43525.695309
28	61	Maschio	16.075990	56055.951355
29	42	Femmina	81.796702	49637.065904
30	66	Maschio	83.213418	36444.472089
31	44	Femmina	50.746773	54865.389226
32	59	Femmina	0.638587	32314.403133
33	45	Maschio	28.703813	67815.190827
34	33	Femmina	61.692692	43030.740541
35	32	Femmina	98.118618	53017.394838
36	64	Maschio	63.181353	54249.318073
37	68	Femmina	25.980358	46116.425394
38	61	Maschio	63.400570	58800.407022
39	69	Femmina	53.998538	42876.445151
40	20	Maschio	77.984540	63069.459462
41	54	Maschio	10.698064	29810.304796
42	68	Femmina	76.102790	51895.693693
43	24	Femmina	54.126658	79083.934986
44	38	Femmina	96.299200	34995.030131
45	26	Femmina	34.187217	39833.825442
46	56	Femmina	63.262189	57708.617745

```

47 35 Maschio 93.202811 52693.726729
48 21 Femmina 10.250973 55259.451489
49 42 Maschio 93.722849 57337.806952
50 31 Maschio 68.788572 59520.822028
51 67 Femmina 6.783706 66645.497629
52 26 Maschio 30.096357 56147.279854

```

Questo codice genera un DataFrame contenente dati casuali su età, genere, punteggio e reddito, quindi stampa le prime 50 righe del DataFrame e successivamente stampa le righe dall'indice 12 all'indice 52 del DataFrame.

```

[9]: import pandas as pd # Importa la libreria pandas per la gestione dei dati
import numpy as np # Importa la libreria numpy per la generazione di dati
    ↪casuali
import matplotlib.pyplot as plt # Importa la libreria matplotlib per la
    ↪visualizzazione dei dati
import seaborn as sns # Importa la libreria seaborn per la visualizzazione
    ↪avanzata dei dati
import plotly.express as px # Importa la libreria plotly express per la
    ↪creazione di grafici interattivi

np.random.seed(42) # Imposta il seme casuale per rendere i risultati
    ↪riproducibili

# Genera dati casuali riguardanti età, genere, punteggio e reddito
data = {
    'Età': np.random.randint(18, 70, size=1000), # Genera età casuali comprese
    ↪tra 18 e 70 anni
    'Genere': np.random.choice(['Maschio', 'Femmina'], size=1000), # Genera
    ↪genere casuali (Maschio o Femmina)
    'Punteggio': np.random.uniform(0, 100, size=1000), # Genera punteggi
    ↪casuali compresi tra 0 e 100
    'Reddito': np.random.normal(50000, 15000, size=1000) # Genera redditi
    ↪casuali con distribuzione normale
}

df = pd.DataFrame(data) # Crea un DataFrame pandas dai dati generati

# Stampa le prime 50 righe del DataFrame
print("Ecco il dataframe che inizia da 0 fino a 49:")
print(df.head(50)) # Prende le prime 50 righe del DataFrame e le stampa

# Stampa le righe dal 13° al 52° del DataFrame
print("Ecco il dataframe che inizia da 12 fino a 52:")
print(df.iloc[12:53]) # Stampa le righe dall'indice 12 all'indice 52 del
    ↪DataFrame

```

I valori mancanti per ciascuna colonna sono:

```

      Età  Genere  Punteggio  Reddito
0    False   False     False    False
1    False   False     False    False
2    False   False     False    False
3    False   False     False    False
4    False   False     False    False
..     ...
995  False   False     False    False
996  False   False     False    False
997  False   False     False    False
998  False   False     False    False
999  False   False     False    False

```

[1000 rows x 4 columns]

I valori mancanti per ciascuna colonna sono:

```

Età          0
Genere       0
Punteggio    0
Reddito      0
dtype: int64

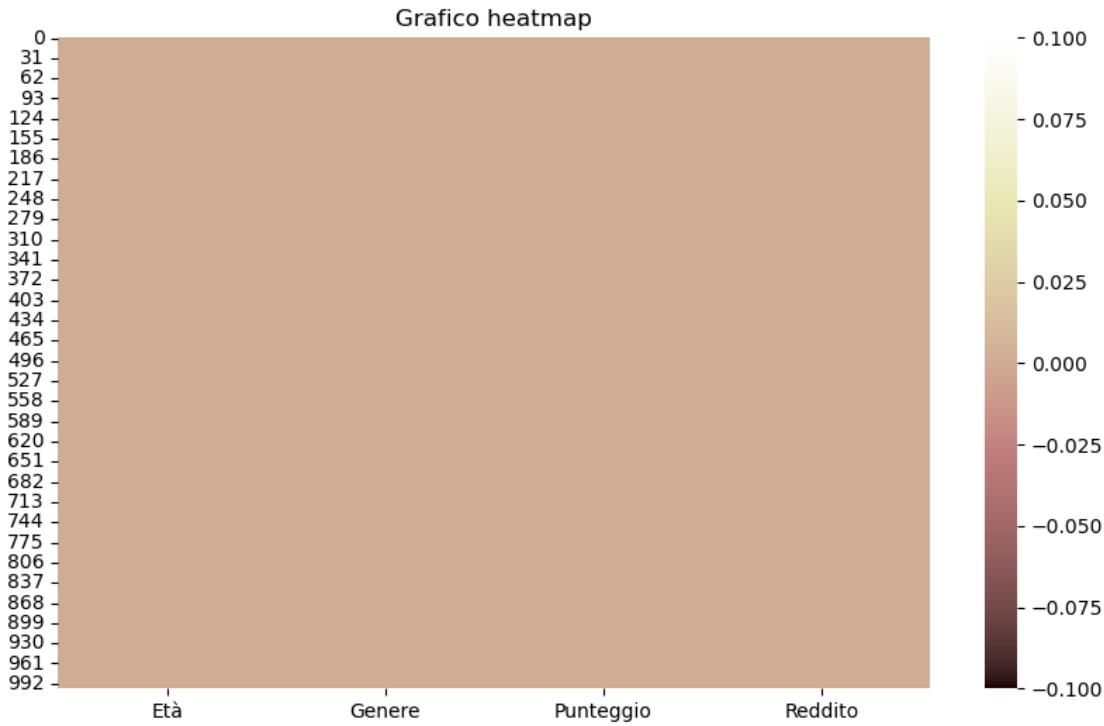
```

Questo codice crea un grafico di tipo heatmap utilizzando Seaborn per visualizzare la distribuzione dei valori nulli nel DataFrame df. La mappa di colori utilizzata è "pink", e la presenza di valori nulli è rappresentata con colori diversi sulla heatmap. La barra dei colori laterale indica la corrispondenza tra i colori e i valori nulli.

```

[12]: plt.figure(figsize=(10, 6)) # Imposta le dimensioni della figura del grafico
sns.heatmap(df.isnull(), cbar=True, cmap="pink") # Crea un grafico a heatmap
# per visualizzare i valori nulli nel DataFrame
# `df.isnull()` restituisce una maschera booleana dove True indica la presenza
# di valori nulli
# `cbar=True` visualizza la barra dei colori laterale per la legenda
# `cmap="pink"` imposta la mappa di colori del grafico a "pink"
plt.title("Grafico heatmap") # Imposta il titolo del grafico
plt.show() # Mostra il grafico

```



Questo codice seleziona le colonne numeriche del DataFrame, quindi crea un istogramma dei valori presenti nella colonna “Punteggio”. La curva KDE (Kernel Density Estimate) è sovrapposta all’istogramma per visualizzare la densità di probabilità stimata. La legenda mostra l’etichetta “Punteggio” e il titolo del grafico è impostato su “Grafico valori”. La dimensione della figura è di 12 pollici di larghezza per 6 pollici di altezza.

```
[6]: import pandas as pd # Importa la libreria pandas con l'alias pd
import numpy as np # Importa la libreria numpy con l'alias np
import matplotlib.pyplot as plt # Importa la libreria matplotlib.pyplot con l'alias plt
import seaborn as sns # Importa la libreria seaborn con l'alias sns

# Genera dati casuali per il DataFrame
np.random.seed(42) # Imposta il seed per la riproducibilità dei dati casuali
data = { # Dizionario contenente i dati casuali
    'Età': np.random.randint(18, 70, size=1000), # Dati casuali per l'età compresi tra 18 e 70 anni
    'Genere': np.random.choice(['Maschio', 'Femmina'], size=1000), # Dati casuali per il genere
    'Punteggio': np.random.uniform(0, 100, size=1000), # Dati casuali per il punteggio compreso tra 0 e 100
    'Reddito': np.random.normal(50000, 15000, size=1000) # Dati casuali per il reddito con distribuzione normale
}
```

```

}

# Creazione del DataFrame 'df' utilizzando i dati generati
df = pd.DataFrame(data)

# Seleziona le colonne numeriche del DataFrame
featuresnumeriche = df.select_dtypes(include=[np.number])

# Imposta le dimensioni della figura del grafico
plt.figure(figsize=(12, 6))

# Imposta lo stile del grafico a "darkgrid" di Seaborn
sns.set_style("darkgrid")

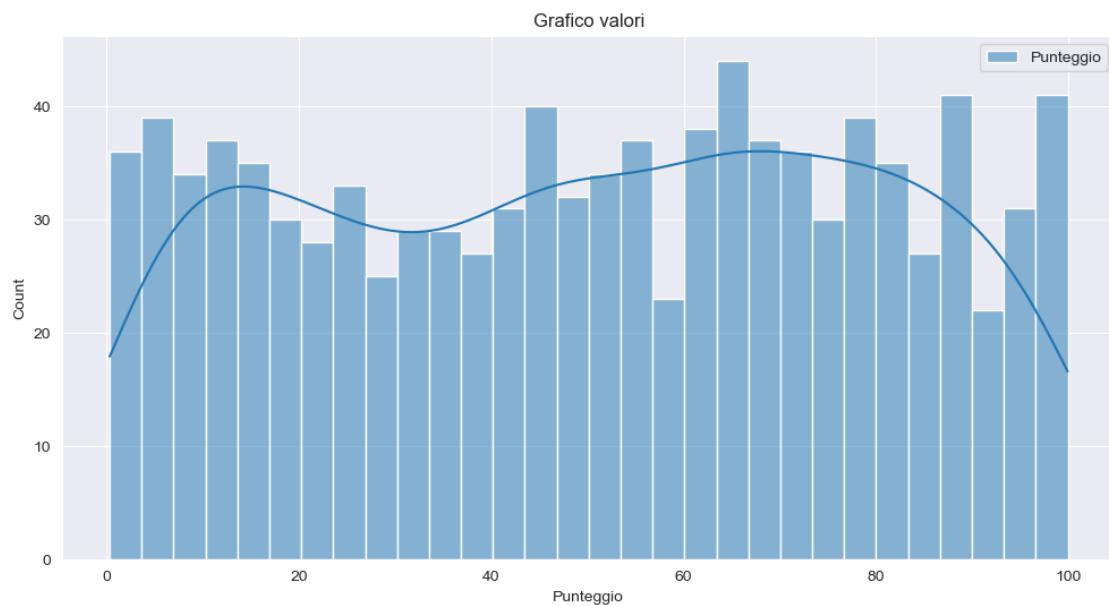
# Crea un istogramma dei valori della colonna "Punteggio"
sns.histplot(df["Punteggio"], kde=True, bins=30, label="Punteggio")

# Mostra la legenda sul grafico
plt.legend()

# Imposta il titolo del grafico
plt.title("Grafico valori")

# Mostra il grafico
plt.show()

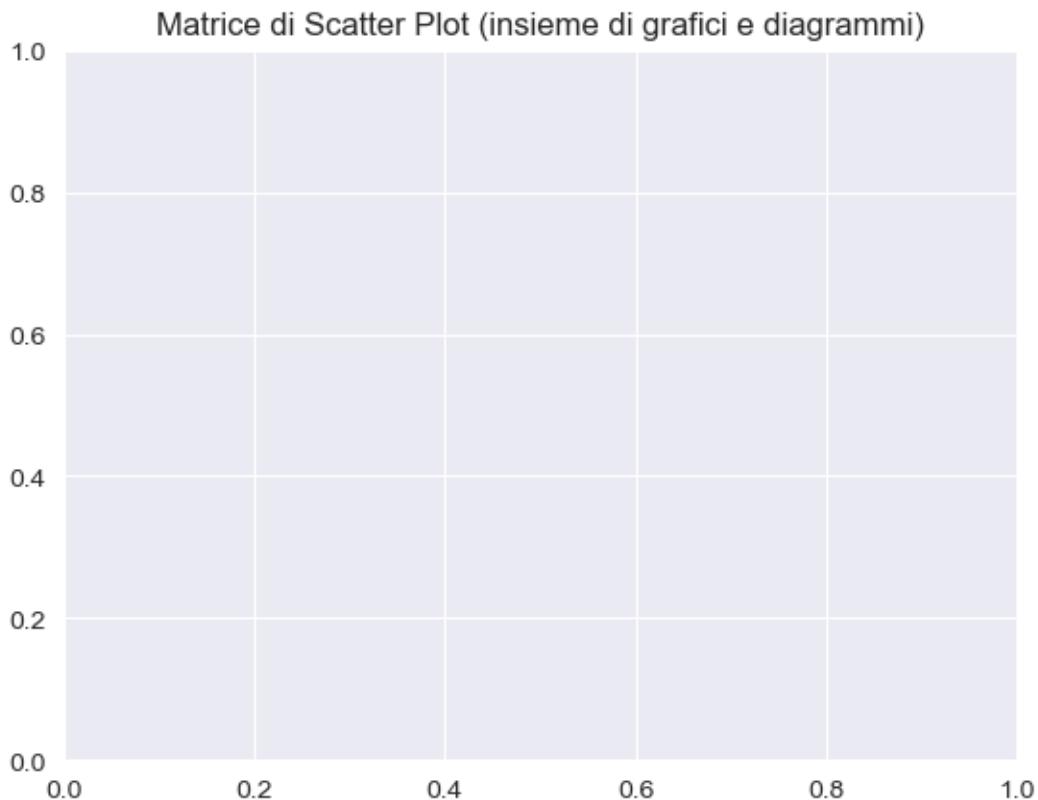
```

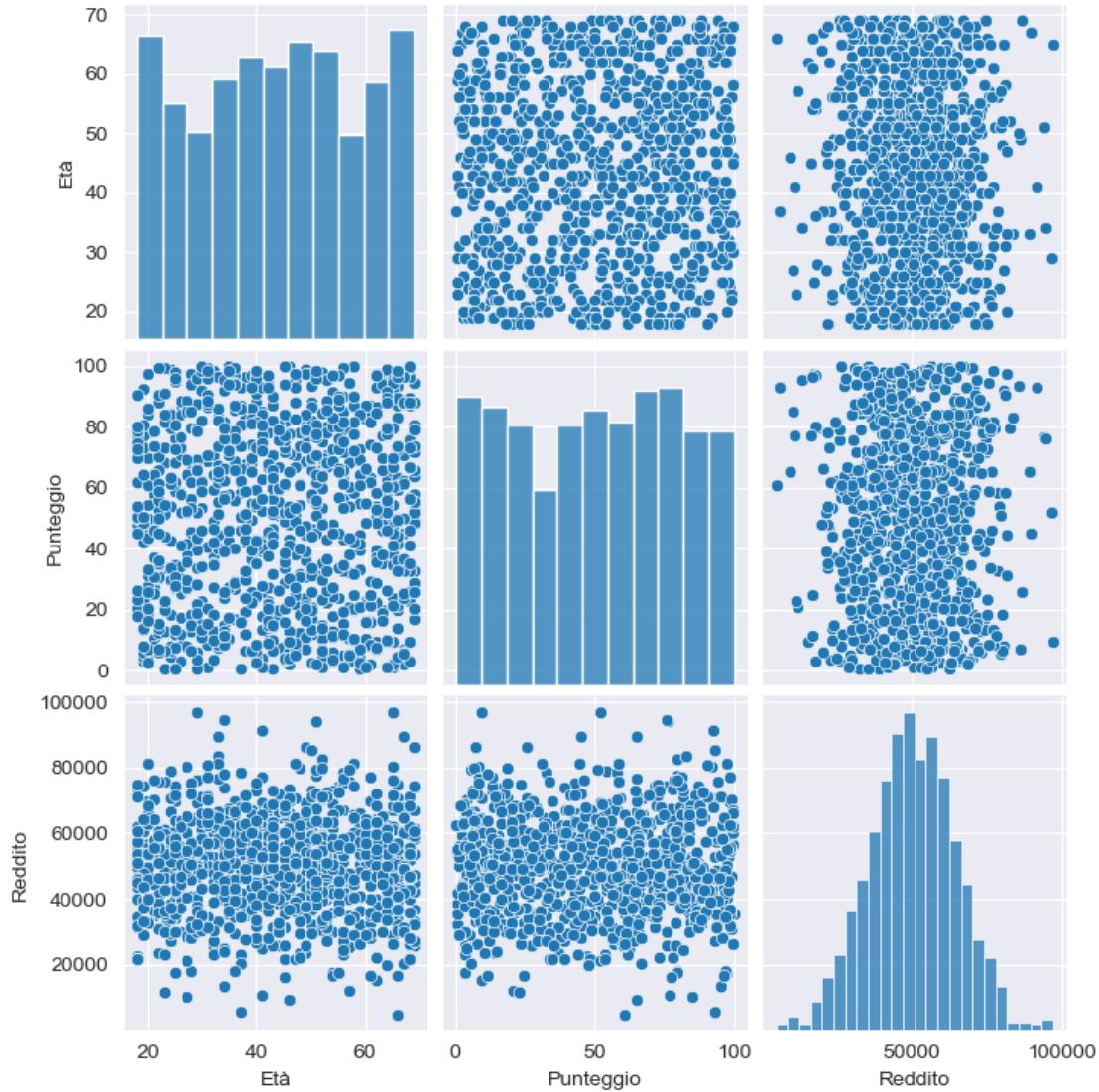


Questo codice traccia una matrice di scatter plot tra tutte le variabili numeriche presenti nel

DataFrame ‘df’, mostrando le relazioni bivariate tra di esse. Ogni cella della matrice rappresenta uno scatter plot tra una coppia di variabili.

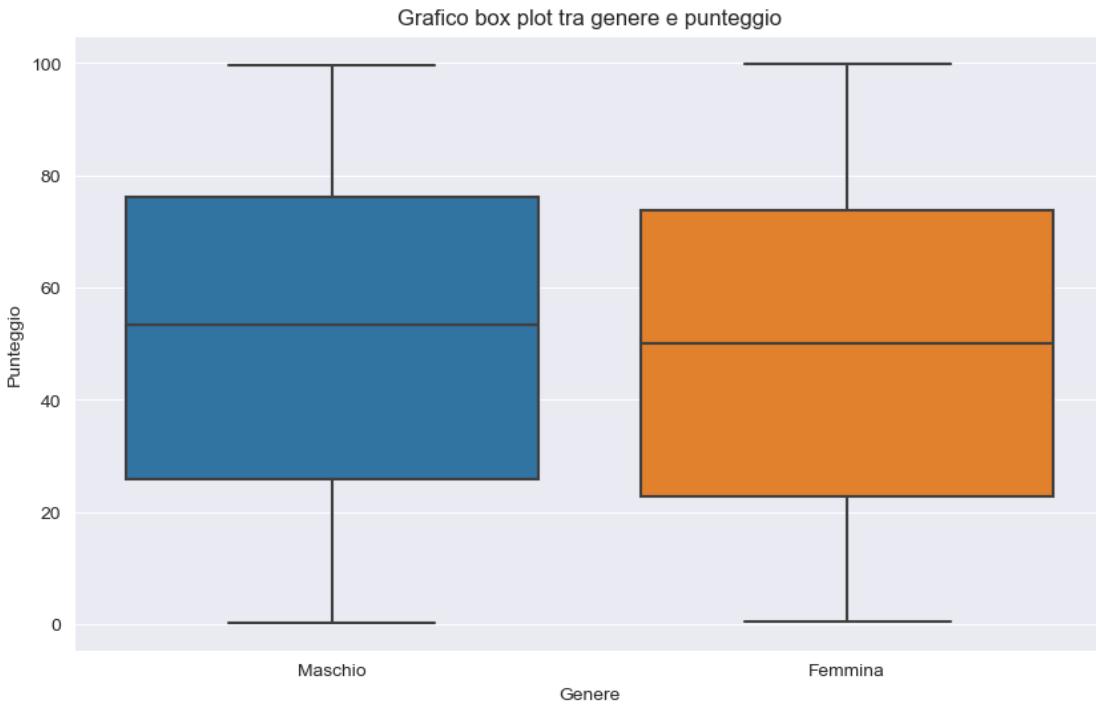
```
[7]: # Visualizza una matrice di Scatter Plot tra le variabili numeriche
plt.title("Matrice di Scatter Plot (insieme di grafici e diagrammi)") # Imposta il titolo del grafico
sns.pairplot(df[featuresnumeriche.columns]) # Crea una matrice di scatter plot per le variabili numeriche del DataFrame
# `df[featuresnumeriche.columns]` seleziona solo le colonne numeriche del DataFrame
plt.show() # Mostra la matrice di scatter plot
```





Questo codice crea un grafico a boxplot per visualizzare la distribuzione dei punteggi in base al genere nel DataFrame 'df'. Il boxplot mostra la mediana, il quartile inferiore e superiore, e gli eventuali outlier per ciascuna categoria di genere.

```
[8]: plt.figure(figsize=(10, 6)) # Imposta le dimensioni della figura del grafico
sns.boxplot(x="Genere", y="Punteggio", data=df) # Crea un boxplot per
# visualizzare la distribuzione del punteggio in base al genere
# `x="Genere"` specifica la variabile categorica da visualizzare sull'asse x
# `y="Punteggio"` specifica la variabile numerica da visualizzare sull'asse y
# `data=df` specifica il DataFrame da cui estrarre i dati
plt.title("Grafico box plot tra genere e punteggio") # Imposta il titolo del
# grafico
plt.show() # Mostra il grafico
```



Il codice genera un dataset casuale con variabili numeriche e categoriche, calcola la matrice di correlazione tra le variabili numeriche e visualizza questa matrice come un heatmap, che mostra visivamente la correlazione tra le varie variabili numeriche.

```
[26]: # Importa le librerie necessarie.
import numpy as np          # libreria per operazioni numeriche e matrici.
import seaborn as sns        # libreria per la visualizzazione avanzata dei dati.
import matplotlib.pyplot as plt # libreria per la creazione di grafici.
#Genera un dataset di esempio con variabili numeriche.
np.random.seed(42) #imposta il seed per la generazione di numeri casuali, ↴ garantendo la riproducibilità.
data = pd.DataFrame(np.random.rand(100, 5),
                     columns=['Variabile1', 'Variabile2', 'Variabile3', ↴
                     'Variabile4', 'Variabile5']) # Crea un DataFrame con 100 righe e 5 colonne di ↴ valori casuali tra 0 e 1.
#Aggiungere alcune variabili categoriche generate casualmente.
data['Categoria1'] = np.random.choice(['A', 'B', 'C'], size=100) # Aggiunge una ↴ nuova colonna 'Categoria1', con valori casuali tra 'A', 'B' e 'C'.
data['Categoria2'] = np.random.choice(['X', 'Y'], size=100) # Aggiunge una ↴ nuova colonna 'Categoria2', con valori casuali tra 'X' e 'Y'.
# Calcola la matrice di correlazione delle variabili numeriche nel DataFrame.
correlation_matrix = data.corr() # Utilizza il metodo 'corr()' per calcolare la ↴ correlazione tra le colonne numeriche.
# Imposta la dimensione della figura per il grafico.
```

```

# Il parametro figsize=(larghezza, altezza) definisce le dimensioni della figura
# in pollici.
plt.figure(figsize=(10, 8))

# Crea un heatmap utilizzando Seaborn.
# 'correlation_matrix' è la matrice di correlazione che vuoi visualizzare.
sns.heatmap(correlation_matrix,
             annot=True, # 'annot=True' aggiunge le etichette (i valori di
# correlazione) in ciascuna cella del heatmap.
             cmap='coolwarm', # 'cmap' definisce la palette di colori da usare.
# 'coolwarm' è una palette da blu (valori bassi) a rosso (valori alti).
             fmt='.2f', # 'fmt' specifica il formato numerico delle etichette.
# Qui si usano due cifre decimali.
             alpha=0.7) # 'alpha' definisce la trasparenza delle celle. 0 è
# completamente trasparente, 1 è completamente opaco.

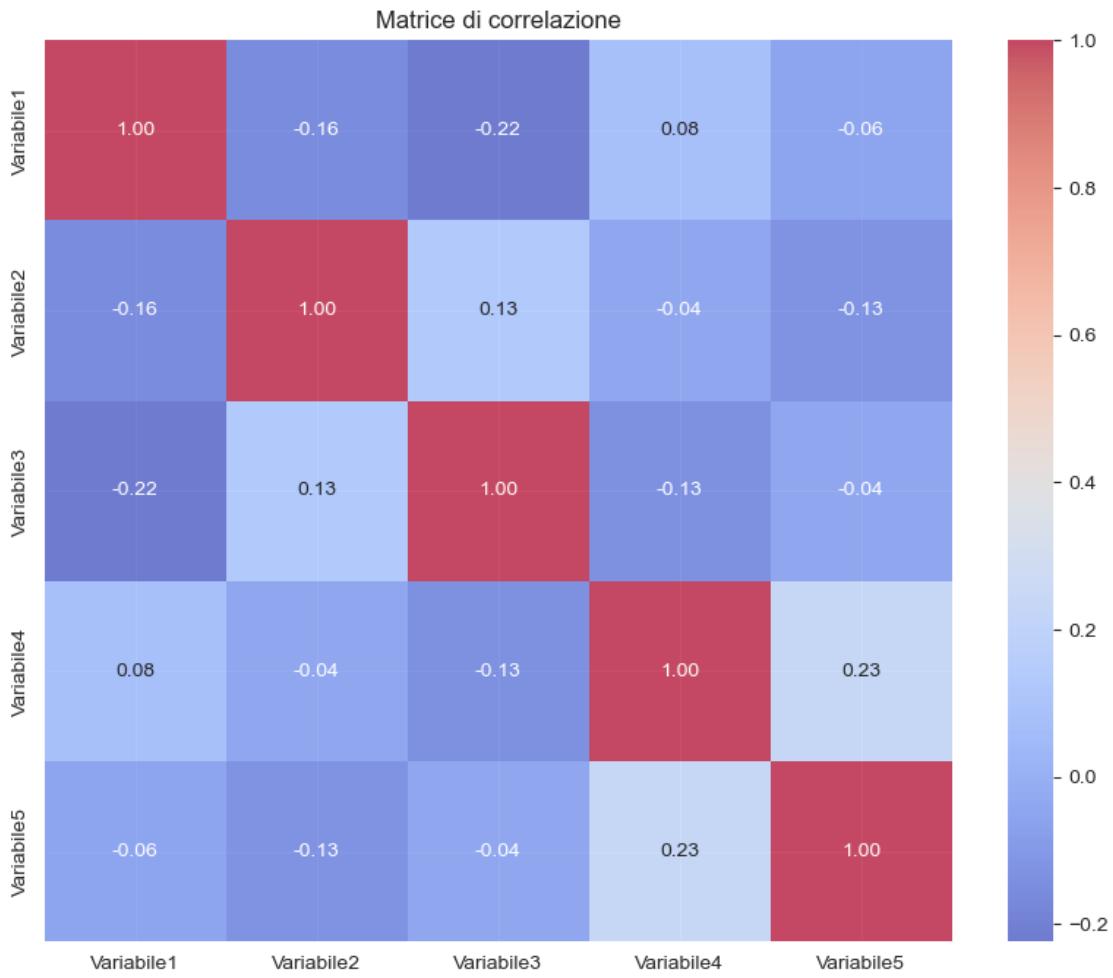
# Imposta il titolo del grafico.
plt.title("Matrice di correlazione")

# Mostra il grafico. Questa funzione rende visibile il grafico a schermo.
plt.show()

```

C:\Users\matte\AppData\Local\Temp\ipykernel_6120\239938224.py:13: FutureWarning:

The default value of numeric_only in DataFrame.corr is deprecated. In a future version, it will default to False. Select only valid columns or specify the value of numeric_only to silence this warning.



Utilizza la libreria pandas per generare un DataFrame casuale con dati categorici e numerici. Vengono create quattro colonne: due colonne di dati categorici (“CatCol1” e “CatCol2”) e due colonne di dati numerici (“NumCol1”, “NumCol2” e “NumCol3”). Successivamente, vengono introdotti valori mancanti in modo casuale nel DataFrame. Infine, il DataFrame risultante viene visualizzato.

```
[27]: import pandas as pd
import numpy as np

# Imposta il seed per la generazione di numeri casuali per rendere i risultati riproducibili.
np.random.seed(41)

# Crea un DataFrame vuoto.
df = pd.DataFrame()

# Genera dati casuali per il DataFrame.
n_rows = 100 # Numero di righe nel DataFrame.
```

```

# Genera una colonna di dati categorici casuali scegliendo tra 'A', 'B', e 'C'.
df['CatCol1'] = np.random.choice(['A', 'B', 'C'], size=n_rows)

# Genera un'altra colonna di dati categorici casuali scegliendo tra 'X' e 'Y'.
df['CatCol2'] = np.random.choice(['X', 'Y'], size=n_rows)

# Genera una colonna di dati numerici da una distribuzione normale standard
# (media 0, deviazione standard 1).
df['NumCol1'] = np.random.randn(n_rows)

# Genera una colonna di dati numerici interi casuali tra 1 e 99 (100 escluso).
df['NumCol2'] = np.random.randint(1, 100, size=n_rows)

# Genera una colonna di dati numerici da una distribuzione uniforme tra 0 e 1.
df['NumCol3'] = np.random.uniform(0, 1, size=n_rows)

# Calcola il numero totale di valori mancanti desiderati nel DataFrame.
total_missing_values = int(0.03 * n_rows * len(df.columns)) # 3% dei valori
# totali.

# Introduce valori mancanti in modo casuale nel DataFrame.
for column in df.columns:
    # Determina casualmente quanti valori mancanti inserire in ogni colonna.
    num_missing_values = np.random.randint(0, total_missing_values + 1)

    # Seleziona indici casuali in cui inserire i valori mancanti.
    missing_indices = np.random.choice(n_rows, size=num_missing_values,
    replace=False)

    # Imposta i valori mancanti (NaN) in quegli indici per la colonna corrente.
    df.loc[missing_indices, column] = np.nan

# Visualizza il DataFrame finale.
df

```

```
[27]:   CatCol1  CatCol2  NumCol1  NumCol2  NumCol3
0        A        X -1.059631    55.0  0.332803
1        A        Y  1.761337    17.0  0.494275
2        C        X  0.165642    52.0  0.378974
3        A        X -0.795811    24.0      NaN
4        B        Y -0.930885     2.0  0.719425
..      ...
95       C        X -1.089337    28.0  0.596135
96       A        Y  1.634932    88.0  0.933752
97       A        Y -2.980924    73.0  0.507170
98       C        X      NaN    93.0  0.272017
```

```
99      B      X  0.375760    44.0  0.853909
```

```
[100 rows x 5 columns]
```

Questo codice seleziona le righe del DataFrame df che contengono almeno un valore mancante, quindi calcola la lunghezza di questo sottoinsieme, che corrisponde al numero di righe con almeno un dato mancante nel DataFrame.

```
[28]: righedatimancanti=df[df.isnull().any(axis=1)]  
len(righedatimancanti)
```

```
[28]: 19
```

Questo codice calcola la percentuale di valori mancanti per ciascuna colonna nel DataFrame.

```
[30]: missingpercent=(df.isnull().sum() / len(df)) * 100  
missingpercent
```

```
[30]: CatCol1    6.0  
CatCol2    0.0  
NumCol1    3.0  
NumCol2    4.0  
NumCol3    8.0  
dtype: float64
```