ESERCIZIO

1. Scrivere una funzione countCharsPerLine(filename) che prende in ingresso una stringa contenente il nome di un file e restituisce una lista di numeri.
Ciascun numero della lista restituita deve corrispondere al numero di carrateri di una linea scritta nel file. Ad esempio, se il file contiene:

```
ciao a tutti
questo è un testo.
```

allora la funzione restituisce la lista:
`[12,18]`

2. Scrivere il codice del blocco "main" che:
- chiede all'utente di inserire il nome del file.
- esegue la funzione countCharsPerLine sul nome del file inserito dall'utente e registra in una variabile il suo risultato. Se la funzione produce un errore, avvisa l'utente.
- scrive sul file "conteggio.txt" una stringa per ogni elemento della lista restituita da countCharsPerLine nel seguente formato:

```
riga 1: 12 caratteri
riga 2: 18 caratteri
```

dove il numero di riga (1, 2) e il numero di caratteri (12, 18) è generato automaticamente per ogni elemento della lista.

####
Provare a comporre la stringa da scrivere sul file usando le seguenti possibilità:
- la sintassi delle f-string di Python3
- il metodo format delle stringhe
Esempio:

```
a=3 #esempio di variabile
b=99  #esempio di variabile

# una f-string si identifica con il carattere f posto prima delle
virgolette.
stringaFString = f"indice {a}, valore {b}"  #La f-string compone
direttamente la stringa sostituendo {a} e {b} con il rispettivo
valore delle variabili a e b.

stringaFormat = "indice {}, valore {}" #il metodo format permette
di specificare in seguito gli elementi da sostituire ai simboli {}

print(stringaFString) # stampo direttamente la stringa
print(stringaFormat.format(a,b)) #uso il metodo format per
specificare cosa sostituire ai simboli {} seguendo l'ordine.
```

In entrambi i casi si ottiene:
`"indice 3, valore 99"`