Esercizi Python 02-05

Segreteria telefonica

Simula una segreteria telefonica semplice. Il programma deve chiedere all'utente di inserire un numero di telefono. Se il numero è presente in un elenco di contatti memorizzati nel programma, deve stampare il nome associato al numero. In caso contrario, deve offrire all'utente la possibilità di aggiungere il nuovo numero e il nome corrispondente all'elenco.

```
def main():
  # Dizionario per memorizzare i contatti (numero: nome)
  contatti = {
    "12345": "Mario Rossi",
    "55555": "Anna Bianchi",
    "98765": "Giovanni Verdi"
  }

  # Richiedere all'utente il numero di telefono
  numero_telefono = input("Inserisci il numero di telefono: ")

  # Controllare se il numero è presente nel dizionario
  if numero_telefono in contatti:
    # Stampare il nome associato al numero
    nome = contatti[numero_telefono]
    print(f"Il numero {numero_telefono} appartiene a {nome}.")
  else:
    # Offrire all'utente la possibilità di aggiungere un nuovo contatto
    aggiungere_contatto = input("Numero non trovato. Vuoi aggiungere un
nuovo contatto (s/n)? ")
    if aggiungere_contatto.lower() == "s":
      nuovo_nome = input("Inserisci il nome del contatto: ")
      contatti[numero_telefono] = nuovo_nome
      print(f"Contatto {nuovo_nome} aggiunto per il numero
{numero_telefono}.")
    else:
      print("Operazione annullata.")

if __name__ == "__main__":
```

```
    main()
```

Ricerca binaria

La ricerca binaria è un algoritmo efficiente per trovare un elemento specifico in una lista ordinata. Funziona suddividendo ripetutamente l'intervallo di ricerca a metà e confrontando l'elemento desiderato con l'elemento nel mezzo della porzione di lista considerata.

1. **Inizializzazione:**
    - L'intervallo di ricerca iniziale include l'intera lista ordinata.
    - Si determina l'elemento di mezzo dell'intervallo di ricerca.
2. **Confronto:**
    - Si confronta l'elemento di mezzo con l'elemento cercato:
        - Se l'elemento di mezzo corrisponde all'elemento cercato, l'algoritmo restituisce l'indicizzazione dell'elemento trovato.
        - Se l'elemento di mezzo è maggiore dell'elemento cercato, si restringe l'intervallo di ricerca alla metà sinistra della lista.
        - Se l'elemento di mezzo è minore dell'elemento cercato, si restringe l'intervallo di ricerca alla metà destra della lista.

```
def ricerca_binaria(lista, elemento):
    sinistra, destra = 0, len(lista) - 1
    while sinistra ≤ destra:
        medio = (sinistra + destra) // 2
        if lista[medio] == elemento:
            return True
        elif lista[medio] < elemento:
            sinistra = medio + 1
        else:
            destra = medio - 1
    return False

nums_ordinati = [1, 3, 5, 7, 9, 11, 13, 15]
elemento_da_cercare = 7
print("L'elemento è presente nella lista:", ricerca_binaria(nums_ordinati,
elemento_da_cercare))
```

Calcolo somma numeri primi

Scrivi una funzione in Python che prenda un intervallo di numeri interi positivi come input e restituisca la somma di tutti i numeri primi presenti nell'intervallo.

```
def sum_of_primes_in_range(start, end):
    total = 0
    for num in range(start, end + 1):
        if num > 1:
            for i in range(2, int(num**0.5) + 1):
                if num % i == 0:
                    break
            else:
                total += num
    return total

start_range = 10
end_range = 20
print("La somma dei numeri primi nell'intervallo è:",
sum_of_primes_in_range(start_range, end_range))
```

Inversione di una stringa

Scrivi una funzione in Python che prenda una stringa come input e restituisca la stringa invertita.

```
def inverti_stringa(stringa):
    return ''.join(stringa[i] for i in range(len(stringa) - 1, -1, -1))

test_string = "Hello, world!"
print("Stringa invertita:", inverti_stringa(test_string))
```