Esercizi di Fondamenti di Informatica

Andrea Gussoni
andrea1.gussoni at polimi.it

Politecnico di Milano

November 28, 2019

Table of Contents

Section 1

Ripasso Esercitazione Precedente

Ripasso

Problemi con esercizi della precedente esercitazione?

- Cassetta degli Attrezzi
- Pari/Dispari
- Fusione Liste
- Matrice Simmetrica

Section 2

Liste

Matrice

Ideare un programma che acquisca da tastiera gli elementi di una matrice $n \times n$. Per farlo, prima l'utente inserirà il valore di n, e successivamente inserirà tutti i valori (con una scansione riga per riga). Una volta acquisita la matrice, il programma dovrà stamparla a schermo, verificare che la matrice sia simmetrica ed in questo caso riportare gli elementi sulle diagonali principale e secondaria.

Matrice I

Una possibile soluzione

```
def main():
    n = -1
    while n < 0:
        n = int(input("Inserire il valore di n"))

    matr = []
    for riga in range(0, n):
        lista_riga = []
        for col in range(0, n):
            print("Inserisci il valore in posizione
                  [", riga, "][", col, "]")
            elem = int(input(""))
            lista_riga.append(elem)
        matr.append(lista_riga)
```

Matrice II

```
print("Matrice inserita:")
for riga in matr:
    print(riga)

simmetrica = True
for r in range(0, n):
    for c in range(0, n):
        if matr[r][c] != matr[c][r]:
            simmetrica = False

if simmetrica == True:
    print("Matrice simmetrica")
    print("Diagonale principale:", end="")
    for i in range(0, n):
        print(matr[i][i], end="")
```

Matrice III

```
        print("\nDiagonale secondaria:", end="")
        for i in range(0, n):
            print(matr[i][n-1-i], end="")
    else:
        print("Matrice non simmetrica")

main()
```

Matrice I

Soluzione ottimizzata

```
def main():
    n = -1
    while n < 0:
        n = int(input("Inserire il valore di n"))

    matr = []
    for riga in range(0, n):
        lista_riga = []
        for col in range(0, n):
            print("Inserisci il valore in posizione
                  [", riga, "][", col, "]")
            elem = int(input(""))
            lista_riga = lista_riga + [elem]
        matr = matr + [lista_riga]
```

Matrice II

```
print("La matrice inserita:")
for riga in matr:
    print(riga)

simmetrica = True
r = 0
c = 0
while r < n and simmetrica == True:
    while c < n and c < r and simmetrica == True:
        if matr[r][c] != matr[c][r]:
            simmetrica = False
        c = c + 1
    r = r + 1

if simmetrica == True:
```

Matrice III

```
        print("Matrice simmetrica")
        print("Diagonale principale:", end="")
        for i in range(0, n):
            print(matr[i][i], end="")
        print("\nDiagonale secondaria:", end="")
        for i in range(0, n):
            print(matr[i][n-1-i], end="")
    else:
        print("Matrice non simmetrica")

main()
```

Alfabeto Farfallino

Ideare un programma che presa in unput una lista formata da caratteri, modifichi la stringa per rispecchiare la codifica ad alfabeto farfallino. La suddetta codifica prevede una trasformazione che in corrispondenza di ogni vocale inserisca una "f" e una ripetizione della vocale stessa (la stringa "a" diventerà "afa").

Alfabeto Farfallino I

Una possibile soluzione

```
def farfallino(l):
    pos = 0
    while pos < len(l):
        elem = l[pos]
        if elem == "a"
        or elem == "e"
        or elem == "i"
        or elem == "o"
        or elem == "u":
            l = l[0:pos] + [elem] + ["f"] + [elem] + l[pos + 1:]
            pos = pos + 3
        else:
            pos = pos + 1
    return l
```

Alfabeto Farfallino II

```
def main():
    lista = ["c", "i", "a", "o"]
    print(farfallino(lista))

main()
```

Alfabeto Farfallino I

Utilizzando una funzione aggiuntiva

```
def vocale(lettera):
    vocali = list("aeiou")
    if lettera in vocali:
        return True
    else:
        return False

def farfallino(l):
    pos = 0
    while pos < len(l):
        elem = l[pos]
        if vocale(elem):
            l = l[0:pos] + [elem] + ["f"] + [elem] + l[pos + 1:]
            pos = pos + 3
```

Alfabeto Farfallino II

```
            else:
                pos = pos + 1
        return l

def main():
    lista = ["c", "i", "a", "o"]
    print(farfallino(lista))

main()
```

Insertion Sort

Ideare un programma che, presa in input una lista di numeri interi, ne effettui l'ordinamento. La parte del programma che effettua l'ordinamento dovrà essere incapsulata in una funzione, in modo da poterla riutilizzare facilmente. L'algoritmo di ordinamento usato sara l'algoritmo di *insertion sort*.

Insertion Sort I

Una possibile soluzione

```
def ordina(lista):

    for i in range(len(lista)):
        elemento = lista[i]
        pos = i

        while pos > 0 and lista[pos - 1] > elemento:
            lista[pos] = lista[pos - 1]
            pos = pos - 1

        # Sistemiamo l'elemento
        lista[pos] = elemento

    return lista
```

Insertion Sort II

```
def main():
    print(ordina([3,2,1]))

main()
```

Section 3

Insiemi e Dizionari

Frequenza Parole

Ideare un programma che, con l'aiuto di un dizionario, stampi un contatore della frequenza delle parole in un testo.

Frequenza Parole I

Una possibile soluzione

```
def main() :
    dizionario_parole = {}

    stringa = """Lorem ipsum dolor sit amet, consectetur adipiscing
              elit , sed do eiusmod tempor incididunt ut labore
              et dolore magna aliqua . Ut enim ad minim veniam,
              quis nostrud exercitation ullamco laboris nisi ut
              aliquip ex ea commodo consequat. Duis aute irure
              dolor in reprehenderit in voluptate velit esse
              cillum dolore eu fugiat nulla pariatur. Excepteur
              sint occaecat cupidatat non proident, sunt in
              culpa qui officia deserunt mollit anim id est
              laborum"""
```

Frequenza Parole II

```
    lista_parole = stringa.split()
    for parola_raw in lista_parole:
        parola = pulisci(parola_raw)
        if parola not in dizionario_parole:
            dizionario_parole[parola] = 1
        else:
            dizionario_parole[parola] = dizionario_parole[parola] + 1

    print("La stringa contiene", len(lista_parole), "parole")
    print("Di cui", len(dizionario_parole), "parole univoche.")
    print(dizionario_parole)

def pulisci(stringa) :
    risultato = ""
    for car in stringa :
```

Frequenza Parole III

```
        if car.isalpha() :
            risultato = risultato + car

    return risultato.lower()

main()
```

Correttore

Ideare un programma che esegua il compito di correttore ortografico, controllando le parole di una certa stringa rispetto alle parole definite in un dizionario di parole corrette.

Anagrammi

Ideare un programma che prenda in unput una parola e cerchi in degli insiemi di anagrammi già definiti tutti i possibili anagrammi di quella parola.

Anagrammi I

Una possibile soluzione

```
def main():
    anagrammi_dizionario = {'acen' : ['ance', 'cane', 'cena'],
                            'aadott' : ['adatto', 'adotta', 'datato'],
                            'ailnop' : ['napoli', 'alpino', 'pilano']}

    parola = input("Inserisci una parola")
    anagrammi = anagrammi_dizionario[''.join(sorted(parola))]

    print("La lista di anagrammi")
    for anagramma in anagrammi:
        print(anagramma)

main()
```

Cruciverba

Ideare un programma che prenda in unput una parola che può contenere una o piu *wildcard*. Il programma deve quindi stampare tutte le parole presenti in un set da utilizzare come dizionario che siano compatibili con la parola inserita.

Cruciverba I

Una possibile soluzione

```
def main():
    inserita = input("Inserisci una parola con eventuali wildcard")
    compatibili = trova_parola(inserita)

    print("Le parole compatibili sono:")
    for compatibile in compatibili:
        print(compatibile)


def trova_parola(parola):
    cesta_parole = set(['cara', 'casa', 'caro', 'naso',
                        'nasa', 'caso', 'cera', 'cero'])
    compatibili = []
```

Cruciverba II

```
    for confronto in cesta_parole:
        if len(confronto) == len(parola):
            uguali = True
            for i in range(len(confronto)):
                if not (parola[i] == confronto[i] or parola[i] == '_'):
                    uguali = False

            if uguali:
                compatibili.append(confronto)

    return compatibili

main()
```

Rubrica Telefonica

Ideare un programma che sfruttando i dizionari Python permetta la consultazione di una semplice rubrica telefonica, includendo anche le funzioni di ricerca per nome, per numero e la stampa di tutti i contatti.

Rubrica Telefonica I

Una possibile soluzione

```
def main() :
    rubrica = {"Anna": 786445,
               "Domenico": 123456,
               "Lilas": 141592,
               "Andrea": 497645 }

    # Cerca Andrea tra i contatti.
    if "Andrea" in rubrica :
        print("Numero di Andrea:", rubrica["Andrea"])
    else :
        print("Andrea non e' presente fra i contatti")

    # Inserisci un nuovo contatto.
    inserito = inserisci_utente(rubrica, "Marco", 763489)
```

Rubrica Telefonica II

```
    if not inserito:
        print("Il contatto era gia' presente")

    # Cerca i contatti con un certo numero di telefono.
    lista_nomi = trova_utente(rubrica, 141592)
    print("Utenti associati a 141592: ", end="")
    for nome in lista_nomi:
        print(nome, end=" ")
    print()

    # Stampa tutta la rubrica.
    stampa(rubrica)

## Trova tutti gli utenti associati ad un certo numero
#  @param rubrica dizionario rappresentante la rubrica
#  @param numero numero da cercare
```

Rubrica Telefonica III

```
#  @return lista degli utenti associati al numero
#
def trova_utente(rubrica, numero) :
    lista = []
    for nome in rubrica:
        if rubrica[nome] == numero:
            lista.append(nome)

    return lista

## Inserisci un nuovo contatto nella rubrica
# @param rubrica dizionario rappresentante la rubrica
# @param nome del contatto da aggiungere
# @param numero del contatto da aggiungere
# @return successo dell'inserimento
#
```

Rubrica Telefonica IV

```
def inserisci_utente(rubrica, nome, numero):
    if nome not in rubrica:
        rubrica[nome] = numero
        print("Inserito il contatto", nome, "con numero", numero)
        return True
    else:
        return False

## Stampa tutti gli elementi della rubrica
#  @param rubrica dizionario rappresentante la rubrica
#
def stampa(rubrica) :
    print("La rubrica e' composta da:")
    for chiave in rubrica:
        print("%-10s %d" % (chiave, rubrica[chiave]))
```

Rubrica Telefonica V

```
main()
```