```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
Created on Fri Jul  2 12:28:35 2021

@author: marco
"""

#Esercizio 1

class Node:                         #Classe nodo
    def __init__(self, data):
        self.right = None
        self.left = None
        self.data = data

    def __str__(self):
        return f'{self.data}'

    def __repr__(self):
        return f'{self.data}'

def insert_a(n,values):               #insert modificato
    if values < n.data:
        if n.left:
            insert_a(n.left,values)
        else:
            n.left=Node(values)
            print("true")
            return

    elif values > n.data:
        if n.right:
            insert_a(n.right,values)
        else:
            n.right=Node(values)
            print("true")
            return
    elif values == n.data:          #controllo se esiste già il valore se vero restituisce false
        print("false")
        return

def insert(n, value):               #insert originale
    if value <= n.data:
        if n.left:
            insert(n.left, value)
        else:
            n.left = Node(value)
    else:
        if n.right:
            insert(n.right, value)
        else:
            n.right = Node(value)
```

```
def populate_tree_from(values, n=None):  #funzione per riempire lalbero con i valori
    print(values)
    if n == None:
        n = Node(values[0])
        values = values[1:]
    for v in values:
            insert(n, v)
    return n

def visit(n):               #stampare l'albero
    if n:
        visit(n.left)
        print(n.data)
        visit(n.right)



##########################################################################################

#Eserizio 2
class Esame:                            #classe Esame
    def __init__(self, materia, cfu, voto):
        self.materia=materia
        self.cfu=cfu
        self.voto=voto


class Studente :                                #classe Studente
    def __init__(self,matricola, nome, cognome, esami):
        self.matricola=matricola
        self.nome=nome
        self.cognome=cognome
        self.esami=esami


    def add_exam(self,mat,c,v):       #aggiungi esame

        for i in self.esami:          #for che scorre la lista esami
            if i.esami != mat:        #controllo se esiste già la materia
                m=Esame(mat,c,v)      #creo un nuovo oggetto esame
                self.esami.append(m)    #lo inserisco nella lista esami
            else:
                return("False")

    def m_pesata(self):       # media pesata
        n=0
        d=0                     #inizializzo il numeratore e il denominatore
        for i in self.esami:        #ciclo che scorre la lista esami

            n += ((i.cfu)*(i.voto))     #sommo il prodotto tra tutti i cfu e i voti
            d += i.cfu                  #sommo tutti i cfu

        media= n/d      #calcolo la media pesata
        return media
```

```
def average_sub(lista,mat):      #media per materia
    media=0
    con=0                        #inizializzo contatore, somma, media
    som=0
    for i in lista:              #for che scorre la lista studenti
        for j in i.esami:        #for che scorre la lista esami di ogni studente
            if mat == j.materia:  #controllo materia da clcolare la media
                con+=1
                som= som + j.voto   #incremento il contatore e faccio la somma dei voti con la materia mat



    media=som/con     #calcolo media
    return(media)

#############################################################################################
#Esercizio 3
def max_m(mat):                  #calcolo il valore massimo nella matrice
    mas=mat[0][0]
    for i in range(len(mat)):
        for j in range(len(mat[0])):
            if mat[i][j] > mas :
                mas = mat[i][j]

    return mas


def min_m(mat):                  #calcolo il valore minimo nella matrice
    mini=mat[0][0]
    for i in range(len(mat)):
        for j in range(len(mat[0])):
            if mat[i][j] < mini :
                mini = mat[i][j]

    return mini

def riscalare(mat):     #matrice riscalata
    matrice_b=mat       #creo una matrice di appoggio di grandezza uguale a quella originale
    val=0               #valore riscalato
    minimo=min_m(mat)       #calcolo il valore minimo nella matrice
    massimo=max_m(mat)      #calcolo il valore massimo nella matrice
    for i in range(len(mat)):               #for che scorrono la matrice
        for j in range(len(mat[0])):
            val=(mat[i][j] - minimo)/(massimo-minimo)     #calcolo valore risclato
            matrice_b[i][j]=val                  # sovrascrivo la matrice b con i valori rscalati
    return matrice_b



#############################################################################################
```

```
if __name__ == "__main__":    #main di partenza
    n = ' '
    root = None

    a=[2, 4, 53, 25, 5, 1]     #lista  valori dell albero

    m=[[1,2],
       [5,0]]                  #matrice da riscalare




    economia=Esame('economia',3,18)
    analisi=Esame('analisi',6,18)
    geometria=Esame('geometria',3,28)     #creo oggetti esame tramite la classe Esame
    analisi2=Esame('analisi',6,20)

    esami1=[economia,analisi]   #lista 1 esami

    esami2=[geometria,analisi2]  #lista 2 esami

    stud1=Studente('7421782','marco','pais',esami1)
    stud2=Studente('1111111','luca','òuca',esami2)          #creo oggetti studenti  tramite la classe Studente e gli passo l
a lista esami


    studenti=[stud1,stud2]    #lista studenti

    while n != '0':      #cilco per effettuare le funzioni
        print('1) esercizio 1 (Albero insert)')
        print('2) esercizio 2A (Media pesata)')
        print('3) esercizio 2B (Aggiungi esame)')                  #opzioni
        print('4) esercizio 2C (Media materia)')
        print('5) esercizio 3 (Matrice riscalata)')
        print('0) esci')
        n = input('la tua scelta corrisponde --> ')

        if n == '1':
            print(a)
            n=int(input("inserisci il valore da inserire -->"))
            albero=populate_tree_from(a)                        #inserisco i valori della lista nel albero
            print(insert_a(albero,n))                  #richiamo la funzione insert per inserire il valore nell albero
            print(visit(albero))              #richiamo la funzione che stampa l'albero
            print("\n")

        elif n == '2':

            print("la media pesata dello studente 1 è", stud1.m_pesata())  #calcolo la media pesata dello studente 1
            print("\n")

        elif n == '3':
            materia=input("inserisci la materia da aggiungere -->")   #inserisco la materia da tastiera
            cfu=int(input("insersci il peso in CFU -->"))           #inserisco i cfu da tastiera
            voto=int(input("insersci il voto -->"))          #inserisco il voto  da tastiera
```

```
        print(stud1.add_exam(materia,cfu,voto))   #rchiamo il metodo add.exam passando le variabile appena inserit
a


        print("\n")

    elif n == '4':
        m=input("inserisci la materia per calcolare la media ") #inserisco la materia da cui calcolare la media
        print(average_sub(studenti,m))      #richiamo la funzione passando la lista studenti e la materia
        print("\n")
    elif n == '5':
        print(m)
        print('Matrice con valori riscalati')
        print(riscalare(m))              #richiamo funzione con la matrice m  e stampo la matrice riscalata
        print("\n")
    elif n == '0':     # se si digita 0 il programma si ferma
        pass
    else:
        print('inserisci un numero esistente ---->'+ n +' non va bene') #controllo se l'utente digita un valore sbagliato
o non esistente
        print("\n")
```