Task:

You are required to develop and test a Python program and then write up a report about your solution.

Students should complete the task based on the scenario chosen and provide evidence to meet all the criteria. Evidence must be provided to show key elements of this assessment:

1. Analysis of the scenario and the application of computational thinking skills.
2. Technical understanding of the data structures, algorithms, functions, and data types.
3. Development of a programming solution in Python.
4. The testing and evaluation of the solution during the development process and at the end of the development.

The following sections should be included in the report submitted:

1. Analysis

2. Design (Using an algorithm)

3. Technical Overview

4. Developing the codded solution

5. Testing to inform development

6. Testing to inform evaluation

7. Evaluation of Solution

8. References

DO NOT COPY

Solution

1. Analysis

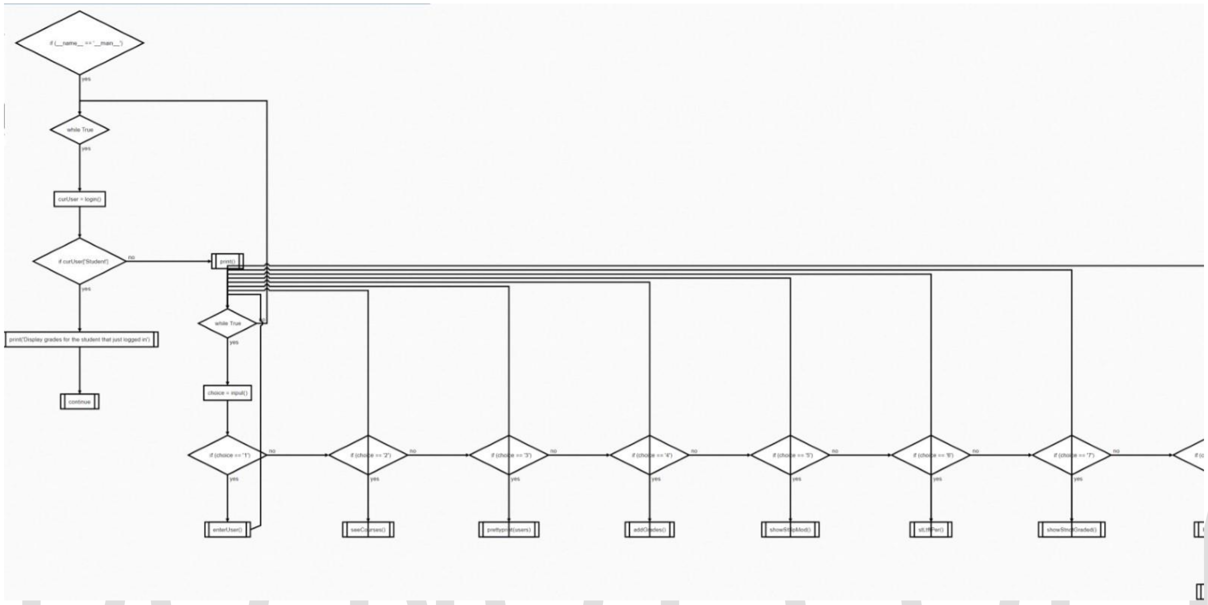
In every educational institution, there are different engagements going on between students and administration. Often these engagements are handled manually and therefore take a significant but unnecessary amount of time because of the advent of different technology related to computers and programming.

Therefore, in this project the basic and primitive working functionalities of student management system are developed with the help of Python programming language in which user will operate using command line interface.

The SMS will have two different roles 1. User and 2. Admin. User will be able to see his/her grades which admin will have different accesses discussed ahead.

SMS should be easy to operate, robust and fast.

2. Design (Using an algorithm)



DO NOT COPY

3. Technical Overview

The programming language used in this project is python. The user engagement will happen through command line interface. Since the approach is procedural and functional programming paradigm, hence at a time single student/admin can use the system.

The initial data about student and admin is stored in the python list first and then dumped into pickle file to be loaded later.

Built in python functions and libraries used:

1. Copy: This library is used to deepcopy "user" dictionary because after assignment of values "user" dictionary would not remain empty. So, to keep the original dictionary unchanged a new copy of same dictionary is leveraged to store corresponding values of a new user.
2. Pickle: It is used to dump or retrieve values to make the runtime values persistent even after the end of runtime.
3. Random: This library is used to get random dummy values to store as grade for a student.
4. PrettyTable: This is used to represent the dictionary values in tabular format.

In the code for loop and while loop has been used for iteration in the functions: login(), showStnotGraded(), stLtftPer(), showStSpMod(), addGrades(), seeCourses(), and enterUser()

Selections are used in the functions: getUser(userName), seeCourses(), showStSpMod(), stLtftPer(), showStnotGraded() and login().

Data structure which is predominantly used is python dictionary which stores details of user, courses and grades. For storing details of multiple users python list is used

Therefore, the dominant variable used in this project is "users" which is a list of dictionaries.

4. Developing the codded solution

```
import copy

import pickle

import random

from prettytable import PrettyTable


# Student Task 3:  Add pickle code to read and write users

# Emtpy template for a  student.


user = {'User Name': '', 'Password': '', 'Name': '', 'Student': True}

courses = {'FC308': {'Ass 1': {'Grade': -1, 'weighting': 0.1},

            'Ass 2': {'Grade': -1, 'weighting': 0.4},

            'Exam': {'Grade': -1, 'weighting': 0.5}},

       'FC315': {'Ass 1': {'Grade': -1, 'weighting': 0.4},

            'Ass 2': {'Grade': -1, 'weighting': 0.6}},

       'FC311': {'Ass 1': {'Grade': -1, 'weighting': 0.1},

            'Ass 2': {'Grade': -1, 'weighting': 0.4},

            'Exam': {'Grade': -1, 'weighting': 0.5}},

       'EXTPRJ': {'Ass 1': {'Grade': -1, 'weighting': 0.4},

             'Ass 2': {'Grade': -1, 'weighting': 0.6}},

       'FC300': {'Ass 1': {'Grade': -1, 'weighting': 0.1},

            'Ass 2': {'Grade': -1, 'weighting': 0.5},

            'Presentation': {'Grade': -1, 'weighting': 0.4}}}
```

DO NOT COPY

```
# Copy and paste additional users here from the console window
# OR generate them by hand from  table above by copy, paste and modifying
# existing users.

users = [{'User Name': 'si2', 'Password': 'SC1000', 'Name': 'Simon Cambers', 'Student': False,
         'Courses': {'FC308': {'Ass 1': {'Grade': 60, 'weighting': 0.1},
                               'Ass 2': {'Grade': 70, 'weighting': 0.4},
                               'Exam': {'Grade': 80, 'weighting': 0.5}},
                     'FC315': {'Ass 1': {'Grade': 59, 'weighting': 0.4},
                               'Ass 2': {'Grade': 89, 'weighting': 0.6}}}},
        {'User Name': 'beanie123', 'Password': 'SG1000', 'Name': 'Steve Gadd', 'Student': True,
         'Courses': {'FC300': {'Ass 1': {'Grade': 27, 'weighting': 0.1},
                               'Ass 2': {'Grade': 40, 'weighting': 0.5},
                               'Presentation': {'Grade': 23, 'weighting': 0.4}},
                     'FC311': {'Ass 1': {'Grade': 34, 'weighting': 0.1},
                               'Ass 2': {'Grade': 10, 'weighting': 0.4},
                               'Exam': {'Grade': 24, 'weighting': 0.5}},
                     'EXTPRJ': {'Ass 1': {'Grade': 34, 'weighting': 0.4},
                                'Ass 2': {'Grade': 25, 'weighting': 0.6}}}},
        {'User Name': 'aisha123', 'Password': 'AK1000', 'Name': 'Aisha Kauser', 'Student': True,
         'Courses': {'FC315': {'Ass 1': {'Grade': -1, 'weighting': 0.4},
                               'Ass 2': {'Grade': -1, 'weighting': 0.6}},
                     'FC300': {'Ass 1': {'Grade': -1, 'weighting': 0.1},
                               'Ass 2': {'Grade': -1, 'weighting': 0.5},
                               'Presentation': {'Grade': -1, 'weighting': 0.4}},
                     'FC311': {'Ass 1': {'Grade': -1, 'weighting': 0.1},
                               'Ass 2': {'Grade': -1, 'weighting': 0.4},
                               'Exam': {'Grade': -1, 'weighting': 0.5}}}},
        {'User Name': 'hashim123', 'Password': 'HA1000', 'Name': 'Hashim Alsadiq', 'Student': True,
```
DO NOT COPY

```
 'Courses': {'FC308': {'Ass 1': {'Grade': -1, 'weighting': 0.1},
                      'Ass 2': {'Grade': -1, 'weighting': 0.4},
                      'Exam': {'Grade': -1, 'weighting': 0.5}},
            'FC315': {'Ass 1': {'Grade': -1, 'weighting': 0.4},
                      'Ass 2': {'Grade': -1, 'weighting': 0.6}},
            'EXTPRJ': {'Ass 1': {'Grade': -1, 'weighting': 0.4},
                       'Ass 2': {'Grade': -1, 'weighting': 0.6}}}},
{'User Name': 'nawaf123', 'Password': 'NA1000', 'Name': 'Nawaf Ali', 'Student': True,
 'Courses': {'FC315': {'Ass 1': {'Grade': -1, 'weighting': 0.4},
                      'Ass 2': {'Grade': -1, 'weighting': 0.6}},
            'FC300': {'Ass 1': {'Grade': -1, 'weighting': 0.1},
                      'Ass 2': {'Grade': -1, 'weighting': 0.5},
                      'Presentation': {'Grade': -1, 'weighting': 0.4}},
            'FC311': {'Ass 1': {'Grade': -1, 'weighting': 0.1},
                      'Ass 2': {'Grade': -1, 'weighting': 0.4},
                      'Exam': {'Grade': -1, 'weighting': 0.5}}}},
{'User Name': 'Jane123', 'Password': 'JS1000', 'Name': 'Jane Smith', 'Student': True,
 'Courses': {'FC300': {'Ass 1': {'Grade': -1, 'weighting': 0.1},
                      'Ass 2': {'Grade': -1, 'weighting': 0.5},
                      'Presentation': {'Grade': -1, 'weighting': 0.4}},
            'FC308': {'Ass 1': {'Grade': -1, 'weighting': 0.1},
                      'Ass 2': {'Grade': -1, 'weighting': 0.4},
                      'Exam': {'Grade': -1, 'weighting': 0.5}},
            'FC311': {'Ass 1': {'Grade': -1, 'weighting': 0.1},
                      'Ass 2': {'Grade': -1, 'weighting': 0.4},
                      'Exam': {'Grade': -1, 'weighting': 0.5}}}},
{'User Name': 'matbat', 'Password': 'mat182', 'Name': 'Mathew', 'Student': True,
 'Courses': {'FC315': {'Ass 1': {'Grade': random.randint(0, 100), 'weighting': 0.4},
                      'Ass 2': {'Grade': random.randint(0, 100), 'weighting': 0.6}},
```

DO NOT COPY

```
            'FC300': {'Ass 1': {'Grade': random.randint(0, 100), 'weighting': 0.1},
                      'Ass 2': {'Grade': random.randint(0, 100), 'weighting': 0.5},
                      'Presentation': {'Grade': random.randint(0, 100), 'weighting': 0.4}},
            'FC311': {'Ass 1': {'Grade': random.randint(0, 100), 'weighting': 0.1},
                      'Ass 2': {'Grade': random.randint(0, 100), 'weighting': 0.4},
                      'Exam': {'Grade': random.randint(0, 100), 'weighting': 0.5}}}},
    {'User Name': 'miller', 'Password': 'millerkiller', 'Name': 'John Miller', 'Student': False,
     'Courses': {'FC315': {'Ass 1': {'Grade': random.randint(0, 100), 'weighting': 0.4},
                      'Ass 2': {'Grade': random.randint(0, 100), 'weighting': 0.6}},
            'FC308': {'Ass 1': {'Grade': random.randint(0, 100), 'weighting': 0.1},
                      'Ass 2': {'Grade': random.randint(0, 100), 'weighting': 0.5},
                      'Presentation': {'Grade': random.randint(0, 100), 'weighting': 0.4}}}},
    {'User Name': 'Nicholas', 'Password': 'nico3402', 'Name': 'Tom Nicholas', 'Student': True,
     'Courses': {'FC315': {'Ass 1': {'Grade': random.randint(0, 100), 'weighting': 0.4},
                      'Ass 2': {'Grade': random.randint(0, 100), 'weighting': 0.6}},
            'FC300': {'Ass 1': {'Grade': random.randint(0, 100), 'weighting': 0.1},
                      'Ass 2': {'Grade': random.randint(0, 100), 'weighting': 0.5},
                      'Presentation': {'Grade': random.randint(0, 100), 'weighting': 0.4}},
            'EXTPRJ': {'Ass 1': {'Grade': random.randint(0, 100), 'weighting': 0.1},
                       'Ass 2': {'Grade': random.randint(0, 100), 'weighting': 0.4},
                       'Exam': {'Grade': random.randint(0, 100), 'weighting': 0.5}}}},
    {'User Name': 'aisha123', 'Password': 'AK1000', 'Name': 'Aisha Kauser', 'Student': True,
     'Courses': {'FC315': {'Ass 1': {'Grade': random.randint(0, 100), 'weighting': 0.4},
                      'Ass 2': {'Grade': random.randint(0, 100), 'weighting': 0.6}},
            'FC300': {'Ass 1': {'Grade': random.randint(0, 100), 'weighting': 0.1},
                      'Ass 2': {'Grade': random.randint(0, 100), 'weighting': 0.5},
                      'Presentation': {'Grade': random.randint(0, 100), 'weighting': 0.4}},
            'FC311': {'Ass 1': {'Grade': random.randint(0, 100), 'weighting': 0.1},
                      'Ass 2': {'Grade': random.randint(0, 100), 'weighting': 0.4},
```

DO NOT COPY

```
                    'Exam': {'Grade': random.randint(0, 100), 'weighting': 0.5}}}},
{'User Name': 'aisha123', 'Password': 'AK1000', 'Name': 'Aisha Kauser', 'Student': True,
 'Courses': {'FC315': {'Ass 1': {'Grade': random.randint(0, 100), 'weighting': 0.4},
                    'Ass 2': {'Grade': random.randint(0, 100), 'weighting': 0.6}},
           'FC300': {'Ass 1': {'Grade': random.randint(0, 100), 'weighting': 0.1},
                    'Ass 2': {'Grade': random.randint(0, 100), 'weighting': 0.5},
                    'Presentation': {'Grade': random.randint(0, 100), 'weighting': 0.4}},
           'FC311': {'Ass 1': {'Grade': random.randint(0, 100), 'weighting': 0.1},
                    'Ass 2': {'Grade': random.randint(0, 100), 'weighting': 0.4},
                    'Exam': {'Grade': random.randint(0, 100), 'weighting': 0.5}}}},
{'User Name': 'aisha123', 'Password': 'AK1000', 'Name': 'Aisha Kauser', 'Student': True,
 'Courses': {'FC315': {'Ass 1': {'Grade': random.randint(0, 100), 'weighting': 0.4},
                    'Ass 2': {'Grade': random.randint(0, 100), 'weighting': 0.6}},
           'FC300': {'Ass 1': {'Grade': random.randint(0, 100), 'weighting': 0.1},
                    'Ass 2': {'Grade': random.randint(0, 100), 'weighting': 0.5},
                    'Presentation': {'Grade': random.randint(0, 100), 'weighting': 0.4}},
           'FC311': {'Ass 1': {'Grade': random.randint(0, 100), 'weighting': 0.1},
                    'Ass 2': {'Grade': random.randint(0, 100), 'weighting': 0.4},
                    'Exam': {'Grade': random.randint(0, 100), 'weighting': 0.5}}}},
{'User Name': 'aisha123', 'Password': 'AK1000', 'Name': 'Aisha Kauser', 'Student': True,
 'Courses': {'FC315': {'Ass 1': {'Grade': random.randint(0, 100), 'weighting': 0.4},
                    'Ass 2': {'Grade': random.randint(0, 100), 'weighting': 0.6}},
           'FC300': {'Ass 1': {'Grade': random.randint(0, 100), 'weighting': 0.1},
                    'Ass 2': {'Grade': random.randint(0, 100), 'weighting': 0.5},
                    'Presentation': {'Grade': random.randint(0, 100), 'weighting': 0.4}},
           'FC311': {'Ass 1': {'Grade': random.randint(0, 100), 'weighting': 0.1},
                    'Ass 2': {'Grade': random.randint(0, 100), 'weighting': 0.4},
                    'Exam': {'Grade': random.randint(0, 100), 'weighting': 0.5}}}},
{'User Name': 'aisha123', 'Password': 'AK1000', 'Name': 'Aisha Kauser', 'Student': True,
```

DO NOT COPY

```
 'Courses': {'FC315': {'Ass 1': {'Grade': random.randint(0, 100), 'weighting': 0.4},
                       'Ass 2': {'Grade': random.randint(0, 100), 'weighting': 0.6}},
             'FC300': {'Ass 1': {'Grade': random.randint(0, 100), 'weighting': 0.1},
                       'Ass 2': {'Grade': random.randint(0, 100), 'weighting': 0.5},
                       'Presentation': {'Grade': random.randint(0, 100), 'weighting': 0.4}},
             'FC311': {'Ass 1': {'Grade': random.randint(0, 100), 'weighting': 0.1},
                       'Ass 2': {'Grade': random.randint(0, 100), 'weighting': 0.4},
                       'Exam': {'Grade': random.randint(0, 100), 'weighting': 0.5}}}},
{'User Name': 'aisha123', 'Password': 'AK1000', 'Name': 'Aisha Kauser', 'Student': True,
 'Courses': {'FC315': {'Ass 1': {'Grade': random.randint(0, 100), 'weighting': 0.4},
                       'Ass 2': {'Grade': random.randint(0, 100), 'weighting': 0.6}},
             'FC300': {'Ass 1': {'Grade': random.randint(0, 100), 'weighting': 0.1},
                       'Ass 2': {'Grade': random.randint(0, 100), 'weighting': 0.5},
                       'Presentation': {'Grade': random.randint(0, 100), 'weighting': 0.4}},
             'FC311': {'Ass 1': {'Grade': random.randint(0, 100), 'weighting': 0.1},
                       'Ass 2': {'Grade': random.randint(0, 100), 'weighting': 0.4},
                       'Exam': {'Grade': random.randint(0, 100), 'weighting': 0.5}}}},
{'User Name': 'aisha123', 'Password': 'AK1000', 'Name': 'Aisha Kauser', 'Student': True,
 'Courses': {'FC315': {'Ass 1': {'Grade': random.randint(0, 100), 'weighting': 0.4},
                       'Ass 2': {'Grade': random.randint(0, 100), 'weighting': 0.6}},
             'FC300': {'Ass 1': {'Grade': random.randint(0, 100), 'weighting': 0.1},
                       'Ass 2': {'Grade': random.randint(0, 100), 'weighting': 0.5},
                       'Presentation': {'Grade': random.randint(0, 100), 'weighting': 0.4}},
             'FC311': {'Ass 1': {'Grade': random.randint(0, 100), 'weighting': 0.1},
                       'Ass 2': {'Grade': random.randint(0, 100), 'weighting': 0.4},
                       'Exam': {'Grade': random.randint(0, 100), 'weighting': 0.5}}}},
{'User Name': 'aisha123', 'Password': 'AK1000', 'Name': 'Aisha Kauser', 'Student': True,
 'Courses': {'FC315': {'Ass 1': {'Grade': random.randint(0, 100), 'weighting': 0.4},
                       'Ass 2': {'Grade': random.randint(0, 100), 'weighting': 0.6}},
```
DO NOT COPY

```
            'FC300': {'Ass 1': {'Grade': random.randint(0, 100), 'weighting': 0.1},
                      'Ass 2': {'Grade': random.randint(0, 100), 'weighting': 0.5},
                      'Presentation': {'Grade': random.randint(0, 100), 'weighting': 0.4}},
            'FC311': {'Ass 1': {'Grade': random.randint(0, 100), 'weighting': 0.1},
                      'Ass 2': {'Grade': random.randint(0, 100), 'weighting': 0.4},
                      'Exam': {'Grade': random.randint(0, 100), 'weighting': 0.5}}}},
    {'User Name': 'aisha123', 'Password': 'AK1000', 'Name': 'Aisha Kauser', 'Student': True,
     'Courses': {'FC315': {'Ass 1': {'Grade': random.randint(0, 100), 'weighting': 0.4},
                      'Ass 2': {'Grade': random.randint(0, 100), 'weighting': 0.6}},
            'FC300': {'Ass 1': {'Grade': random.randint(0, 100), 'weighting': 0.1},
                      'Ass 2': {'Grade': random.randint(0, 100), 'weighting': 0.5},
                      'Presentation': {'Grade': random.randint(0, 100), 'weighting': 0.4}},
            'FC311': {'Ass 1': {'Grade': random.randint(0, 100), 'weighting': 0.1},
                      'Ass 2': {'Grade': random.randint(0, 100), 'weighting': 0.4},
                      'Exam': {'Grade': random.randint(0, 100), 'weighting': 0.5}}}},
    {'User Name': 'aisha123', 'Password': 'AK1000', 'Name': 'Aisha Kauser', 'Student': True,
     'Courses': {'FC315': {'Ass 1': {'Grade': random.randint(0, 100), 'weighting': 0.4},
                      'Ass 2': {'Grade': random.randint(0, 100), 'weighting': 0.6}},
            'FC300': {'Ass 1': {'Grade': random.randint(0, 100), 'weighting': 0.1},
                      'Ass 2': {'Grade': random.randint(0, 100), 'weighting': 0.5},
                      'Presentation': {'Grade': random.randint(0, 100), 'weighting': 0.4}},
            'FC311': {'Ass 1': {'Grade': random.randint(0, 100), 'weighting': 0.1},
                      'Ass 2': {'Grade': random.randint(0, 100), 'weighting': 0.4},
                      'Exam': {'Grade': random.randint(0, 100), 'weighting': 0.5}}}},
    {'User Name': 'aisha123', 'Password': 'AK1000', 'Name': 'Aisha Kauser', 'Student': True,
     'Courses': {'FC315': {'Ass 1': {'Grade': random.randint(0, 100), 'weighting': 0.4},
                      'Ass 2': {'Grade': random.randint(0, 100), 'weighting': 0.6}},
            'FC300': {'Ass 1': {'Grade': random.randint(0, 100), 'weighting': 0.1},
                      'Ass 2': {'Grade': random.randint(0, 100), 'weighting': 0.5},
```

DO NOT COPY

```
                'Presentation': {'Grade': random.randint(0, 100), 'weighting': 0.4}},
           'FC311': {'Ass 1': {'Grade': random.randint(0, 100), 'weighting': 0.1},
                'Ass 2': {'Grade': random.randint(0, 100), 'weighting': 0.4},
                'Exam': {'Grade': random.randint(0, 100), 'weighting': 0.5}}}}
     ]


# store users data
f = open('studManSys.bin', 'wb')
pickle.dump(users, f)
f.close()


# read the users data
f = open('studManSys.bin', 'rb')
users = pickle.load(f)
f.close()
assert len(users) == 20, "Users not loaded successfully"


#Enter details of a user
def enterUser():
  while True:
    newCourse = {}
    newUser = copy.deepcopy(user)
    newUser['User Name'] = input('Enter user name: ')
    newUser['Password'] = input('Enter password: ')
    newUser['Name'] = input('Enter name: ')
    if input('(S)tudent/(T)utor?: ') == 'T':
      newUser['Student'] = False
```

DO NOT COPY

```
        while True:
            course = input('Enter course title (Type \'end\' when finished): ')
            if course.lower() != 'end':
                newCourse[course] = (courses[course])
            else:
                break
        print(newCourse)  # debug
        newUser['Courses'] = newCourse
        users.append(newUser)
        # print(users)
        if input('Enter another user (y/n)? ') == 'n':
            break


# Returns the record of the specified Student
def getUser(userName):
    for i in range(len(users)):
        if users[i]['User Name'] == userName:
            return users[i], True
    return {}, False



# See courses of the student
def seeCourses():
    while True:
        user = {}
        userFound = False
```

DO NOT COPY

```
        print('See Courses')
        userName = input('Enter student user name: ')
        if userName == 'end':
            break
        user, userFound = getUser(userName)
        if userFound:
            courses = user['Courses']
            print(courses)
        else:
            print('user not found')


# Add Grades to the student
def addGrades():
    while True:
        print('Add Grades')
        userName = input('Enter student user name: ')
        if userName == 'end':
            break
        user, userFound = getUser(userName)
        if userFound:
            for c in user['Courses']:
                for a in user['Courses'][c]:
                    print(f'enter grade for {a} on {c}: ', end='')
                    g = int(input())
                    user['Courses'][c][a]['Grade'] = g
            print(user['Courses'])
        else:
            print('user not found')
```

DO NOT COPY

```
# show Student under specified module
def showStSpMod():
    choice = int(input(
        'Choose the module for which you want to see students:\n[1]: Ass 1\n[2]: Ass 2\n[3]: Exam\n[4]:
Presentation\n'))
    module = ""
    if choice == 1:
        module = "Ass 1"
    elif choice == 2:
        module = "Ass 2"
    elif choice == 3:
        module = "Exam"
    elif choice == 4:
        module = "Presentation"

    print("Students in this chosen module: ")
    for user in users:
        for course in user['Courses']:
            try:
                if user['Courses'][course]['Exam'] and user['Student'] == True:
                    print(user['Name'])
            except:
                break


# show student having grades less than 40%
```

DO NOT COPY

```
def stLtftPer():
    for user in users:
        if user["Student"]:
            name = user["Name"]
            courses = user["Courses"]
            total_weighted_grade = 0
            total_weighting = 0
            for course, assessments in courses.items():
                course_weighted_grade = 0
                course_weighting = 0
                for assessment, grade_weighting in assessments.items():
                    grade = grade_weighting["Grade"]
                    weighting = grade_weighting["weighting"]
                    if grade >= 0:
                        course_weighted_grade += grade * weighting
                        course_weighting += weighting
                if course_weighting > 0:
                    course_average = course_weighted_grade / course_weighting
                    total_weighted_grade += course_average * course_weighting
                    total_weighting += course_weighting
            if total_weighting > 0:
                overall_average = total_weighted_grade / total_weighting
                if overall_average < 40:
                    print(name)



# Show student not graded yet
def showStnotGraded():
```

DO NOT COPY

```
    for user in users:
        if user["Student"]:
            name = user["Name"]
            courses = user["Courses"]
            total_weighted_grade = 0
            total_weighting = 0
            for course, assessments in courses.items():
                course_weighted_grade = 0
                course_weighting = 0
                for assessment, grade_weighting in assessments.items():
                    grade = grade_weighting["Grade"]
                    weighting = grade_weighting["weighting"]
                    course_weighted_grade += grade * weighting
                    course_weighting += weighting
                if course_weighting > 0:
                    course_average = course_weighted_grade / course_weighting
                    total_weighted_grade += course_average * course_weighting
                    total_weighting += course_weighting
            if total_weighting > 0:
                overall_average = total_weighted_grade / total_weighting
                if overall_average < 0:
                    print(name)


# Login a user
def login():
    while True:
        userFound = False
```

DO NOT COPY

```
        currentUser = {}
        logName = input('Enter login name: ')
        for i in range(len(users)):
            if users[i]['User Name'] == logName:
                currentUser = users[i]
                userFound = True
        if userFound:
            logPass = input('Enter Login password: ')
            break
        else:
            print('user not found')
    return (currentUser)



# logout a user
def logout():
    # save the latest database
    f = open('studManSys.bin', 'wb')
    pickle.dump(users, f)
    f.close()



# print user in columnar format
def prettyprint(users):
    table = PrettyTable()
    table.field_names = ["User Name", "Name", "Student", "Courses"]
    for user in users:
        courses = user["Courses"]
```

DO NOT COPY

```
        courses_str = "\n".join([f"{course}: {values}" for course, values in courses.items()])
        table.add_row([user["User Name"], user["Name"], user["Student"], courses_str])
    print(table)



# Start of code here. User Logs in. After succesful login.


# if teacher then Displays menu and facilitates use of functions above


# else if student dispay grades for that student


while True:
    curUser = login()
    if curUser['Student']:
        # Student Task 1: Display the grades for student logged in.
        print('Display grades for the student that just logged in')
        continue
    else:
        print()
        while True:
            print('---- Student Management System ----')
            print('''
            Press 1 to enter new student.
            Press 2 to see a students courses.
            Press 3 to show all users.
            press 4 to add student grades.
```

DO NOT COPY

```
press 5 to see Students on a specified module.

press 6 to see students under 40% grade average.

press 7 to show students not yet graded.

Press 8 to logout.

''')

choice = input()

if choice == '1':

   enterUser()

elif choice == '2':

   seeCourses()

elif choice == '3':

   # Student Task 2: extract from 'users' the 'user name' and actual name

   # and display neatly in columns

   prettyprint(users)

elif choice == '4':

   addGrades()

elif choice == '5':

   showStSpMod()

elif choice == '6':

   stLtftPer()

elif choice == '7':

   showStnotGraded()

elif choice == '8':

   logout()

   break

else:

   print('Invalid input \n')
```

DO NOT COPY

5. Testing to inform development

For checking whether the user data is loaded correctly or not an assert statement is added into the code like below.
assert len(users) == 20, "Users not loaded successfully"

Primarily we are starting with 20 users only. So we are checking whether details of all 20 users are loaded or not.

6. Testing to inform evaluation

We have added different assert statements to check the results of the functions mentioned below.

#test functions:

#enterUser(). Enter below details:

'''

Enter user name: BobW

Enter password: bob123

Enter name: Bob Walker

(S)tudent/(T)utor?: S

Enter course title (Type 'end' when finished): FC308

Enter course title (Type 'end' when finished): end

{'FC308': {'Ass 1': {'Grade': -1, 'weighting': 0.1}, 'Ass 2': {'Grade': -1, 'weighting': 0.4}, 'Exam': {'Grade': -1, 'weighting': 0.5}}}

Enter another user (y/n)? n

'''

enterUser()

assert getUser('BobW')==({'User Name': 'BobW', 'Password': 'bob123', 'Name': 'Bob Walker', 'Student': True, 'Courses': {'FC308': {'Ass 1': {'Grade': -1, 'weighting': 0.1}, 'Ass 2': {'Grade': -1, 'weighting': 0.4}, 'Exam': {'Grade': -1, 'weighting': 0.5}}}}, True),"enterUser() failed"

#Enter "BobW" as username for below function and check if result is same as below:

#{'FC308': {'Ass 1': {'Grade': -1, 'weighting': 0.1}, 'Ass 2': {'Grade': -1, 'weighting': 0.4}, 'Exam': {'Grade': -1, 'weighting': 0.5}}}

seeCourses()

#stLtftPer() should show Bob Walker in it

DO NOT COPY

stLtftPer()

7. Evaluation of Solution

The solution is done according to the selected functionalities. The implementations are correct and exception handling is done wherever required. Test cases are passing for both development and evaluation perspective. The system is easy to operate, and a proper menu driven approach has been implemented. There is scope of a few more functionalities into the system.

8. References

- Gomathy, C K. (2022). STUDENT INFORMATION MANAGEMENT SYSTEM. INTERANTIONAL JOURNAL OF SCIENTIFIC RESEARCH IN ENGINEERING AND MANAGEMENT. 06. 10.55041/IJSREM11816.
- Fundamentals of project management for development organization, 2 nd edition, PDEVM, Project Management for Development Organization, pp. 13-20.

DO NOT COPY