Health Management System

ICS 104 Lab Project - Term 212

Deadline: Friday May 6 (Before midnight)

1- Description:

You are to develop a **health management system** that tracks the weight of patients visiting **KFUPM Clinic**. The system administrator should be able to view the weight, the average weights and the number of visits of each patient. The system should have the following functions: (all the functions are explained in this document).

- **main( )**
- **getWeight(weight)**
- **getAllPatientRecord( )**
- **getPatient(patientID)**
- **update(patientID, weight)**
- **addPatient(Name, patientID)**
- **deletePatient(patientID)**

Implement a well-structured Python program that uses the functions listed above. The system should enable the **KFUPM Clinic's System Administrator** to maintain Patient's weight. The information is kept in a text-file of the form:

Patient ID	Patient Name	Weight	Avg.Weight	Visits
1007	Ali Ahmed	51.5	51.5	1
1004	Abeer Hassan	80.1	80.1	1
1003	Sameer Bander	25.5	28.5	2
1002	Ibrahim Majed	18 .6	18.2	2
1006	Muhammad Ali	81.2	81.2	1
1005	Taher Mohammed	86.7	88.7	2
1001	Hana Farooq	10.1	10.1	1

Where each line of the text-file contains a **unique** patient ID, the patients' first and second names, his/her weight during the visit, his/her average weight and their number of visits.

The **main ( )** program must have the following main menu:

1. Display all Patient Record
2. Display the Record of a particular Patient
3. Display all Patient Weight
4. Update Patient
5. Add New Patient
6. Delete Patient
0. Exit
Please select your choice:

Your program must loop as long as the option 0 has not been selected. It must display an appropriate error message if an invalid choice is entered. Each of the options 1 to 6 must be implemented in a separate function as mention earlier. The options must have the following behaviors:

Option 1: Display all Patient Record

It displays all the Patients' information recorded. The option must be implemented by reading directly from the text-file where the data was saved. It then waits for the Enter key to be pressed before returning control to the main menu. Below is a sample run:

```
Please select your choice: 1
Patient ID   Patient Name      Weight   Avg.Weight   Visits
1007         Ali Ahmed         51.5     51.5         1
1004         Abeer Hassan      80.1     80.1         1
1003         Sameer Bander     25.5     28.5         2
1002         Ibrahim Majed     18 .6    18.2         2
1006         Muhammad Ali      81.2     81.2         1
1005         Taher Mohammed    86.7     88.7         2
1001         Hana Farooq       10.1     10.1         1
Press Enter key to continue . . .
```

Option 2: Display the Record of a particular Patient

When a user selects option 2, the main function prompts for and reads a **PatientID**. Then it calls the function **getPatient(patientID)**, which then searches for this **PatientID** in the text-file. If the Patient is not found, an appropriate error message is displayed, otherwise; the Patient's information is displayed. In both cases, the option waits for the Enter key to be pressed before returning control to the main menu.

```
Please select your choice: 2
Enter Patient ID: 1002

Patient ID   Patient Name      Weight   Avg.Weight   Visits
1002         Ibrahim Majed     18 .6    18.2         2

Press Enter key to continue . . .
```

```
Please select your choice: 2
Enter Patient ID: 1552

Error: Invalid Patient
Press Enter key to continue . . .
```

Option 3. Display all Patient Weight

It prompts for- and reads a maximum weight from the user. Then it displays all patients with weight less than or equal to the given weight. Also, it displays the percentage weight increase (with a + sign) or decrease (with a - sign) in the patients' weight from the average weight, using the formula in Eqn (1) below:

$$P = (\text{weight} - \text{average weight}) / (\text{average weight}) \times 100 \quad \ldots\ldots\ldots\ldots\ldots (1)$$

Please select your choice: **3**
Please enter max. weight (Kg): **55**

Patient ID	Patient Name	Weight	Avg.Weight	P(%)	Visits
1007	Ali Ahmed	51.5	51.5	0.00	1
1003	Sameer Bander	25.5	28.5	-10.53	2
1002	Ibrahim Majed	18 .6	18.2	+2.197	2
1001	Hana Farooq	10.1	10.1	0.00	1

Press Enter key to continue . . .

Please select your choice: **3**
Please enter max. weight (Kg): **- 55**

Error: Invalid weight

Press Enter key to continue . . .

Options 4. **Update Patient**

This option uses the function **update (patientID, weight)**. It prompts for and reads a **PatientID**, and their current weight (**weight**). The new average weight can be calculated using Eqn (2) and the new visit counts can be calculated using Eqn (3).

Avg. weight = (weight + avg. weight * visit) / (visit + 1) …….. (2)

visit = visit + 1 ……..…. (3)

If the **PatientID** is zero, or the **PatientID**, **visits** and/or any of the weights (i.e., **weight and average weight** is negative) an appropriate error message is displayed; otherwise it searches for this **PatientID** in the text-file and update their record as described earlier. Also, if the **PatientID** is not found an appropriate error message is displayed, as shown by the test run below:

```
Please select your choice: 4
Please enter Patient ID: 1001
Please enter current weight (Kg): 11.15
Please enter average weight (Kg): 10.25
Please enter the patients number of visits: 3

Patient's information has been updated . . .
Press Enter key to continue . . .
```

```
Please select your choice: 4
Please enter Patient ID: 2256
Please enter current weight (Kg): 11.15
Please enter average weight (Kg): 10.25
Please enter the patients number of visits: 3

Error: Invalid patient ID
Press Enter key to continue . . .
```

```
Please select your choice: 4
Please enter Patient ID: 1001
Please enter current weight (Kg): -11.15
Please enter average weight (Kg): 10.25
Please enter the patients number of visits: 3

Error: Invalid weight
Press Enter key to continue . . .
```

Control is returned to the main menu after pressing the Enter key.

You can display the patient record again after updating the patient's information to make sure that the patient's information has been updated successfully as shown below. [Notice that the new Avg.Weight =10.47].

```
Please select your choice:  2
Enter Patient ID: 1001

   Patient ID    Patient Name      Weight      Avg.Weight      Visits
      1001       Hana Farooq       11.15         10.47          3
Press Enter key to continue . . .
```

Options 5. Add New Patient

To implement Option 5, search the text-file for the **PatientID** of the record to be added. If the **PatientID** exists, display an error; otherwise append the new record to the end of the text-file.

Options 6. Delete Patient

To implement option 6, search the text-file for the **PatientID** and deleted the patient when found. If the **PatientID** does not exist, display an error.

2- Guidelines:

1- The lab project should include the following items:

(a) Dealing with diverse data type like strings, floats and int.

(b) Involving operations dealing with files (reading from files and writing to files).

(c) Using Lists/Dictionaries/Sets/Tuples (any of these data structures or combination).

(d) Adding, removing, and modifying records.

(e) Sorting data based on a certain criteria (if it is needed).

2- The lab project will be done by teams of 2 students. However, each student has to know about all tasks in the project and both students will be asked about the project individually.

3- The students should know the following items:

(a) Comments are important; they are worth. **(Worth 5%)**

(b) The code must use meaningful variable names and modular programming **(Worth 10%)**

(c) Global variables are not allowed. Students should learn how to pass parameters to functions and receive results.

(d) Students must submit a working program. Non-working parts can be submitted separately. If a team submits a non-working program, **it loses 20% of the grade**.

(e) Your program should not crash, you need to handle invalid inputs properly. (i.e. valid range and valid type).

4- The students must not share code with another project group. Doing so will result in a zero grade for all groups involved.

5- You are limited to the material covered in the course lectures and labs in your code. Using uncovered material/libraries is not allowed.

6- The **deadline** for submitting the lab project is **Friday May 6 before midnight**.

7- Submission will be through the blackboard only – assignment section (you have only two attempts) and one submission per team is enough.

3- Deliverable:

1- Each team has to submit:

(a) The **code** as a **Jupyter notebook** (**SecXX-GroupXX.ipynb**). Your **jupyter file** should have the following lines at the beginning (I encourage every one of you to contribute equally to the project):

Name of First Student, **ID, SN, Sec-No**.

Name of Second Student, **ID, SN, Sec-No**.

(b) The **report** as a **word/PDF file (Project_Document_Template)**. The report will describe how you solved the problem. In addition, you need to describe the different functions with their tasks and screen shots of their outputs. **(Worth 20%)**

Note: The submission should **contain all the necessary files, code and documentation** that will help test the project. ***The submitted zip file must be in the format***: **SecXX-GroupXX.zip.**

2- Each team has to do Lab demo/presentation:

(a) **Week fifteen** will be used for lab project demos during the lab period.

(b) A slot of around 10-15 minutes will be allocated to each team for their presentation and questions (I will announce it later on in sha Allah).

(c) Students who **do not appear for lab demo/presentation** will **get zero** in the project.

Note that, **35% of the grade** are **highlighted above** and the **remaining 65%** will be on the **code itself and the demo/presentation.**