Transcript Generation System
ICS 104 Lab Project - Term 212
Prepared by: Ashraf Saeed
Deadline: End of Week-14

Develop a menu driven complete program that can perform the illustrated features in Figure 1. This program works for an individual user (i.e. student) only; it can generate the transcript of the student in a txt-file format and it shows some statistics.

Student Transcript Generation System

===
1. Student details
2. Statistics
3. Transcript based on major courses
4. Transcript based on minor courses
5. Full transcript
6. Previous transcript requests
7. Select another student
8. Terminate the system

===
Enter Your Feature:

Figure 1: Transcript generation system menu.

1   Description:

The program starts by asking the user for the student ID (**stdID**) (i.e. 202006000).Note that,If the user enters a wrong ID, the program will keep asking him for an acceptable student ID based on the available IDs in the database. Once he entered an acceptable stdID, the program will show the available degree(s) for this student (i.e. Bachelor (**BS**), Master (**M**), Doctorate (**D**)). The user can select what he wants and he is also required to store the the selected option(s) to generate some services from the menu. Next, the system menu will appear for the user same as in Figure 1.

1.1   Functions:

Each feature has to be designed using a separate function and it might be called by other functions including the **main** function. Note that, you need to pass variables in your functions and you are not allowed to use global variables. Also, the variables and the number of variables that you are going to use should be defined based on the purpose of the function. The descriptions of the start function, and other functions are given below:

1. **Start Feature – def startFeature**():

   This function allows the user to enter the student ID (**stdID**) (i.e. 202006000) and the required degree(s).Note that,If the user enters a wrong ID, the program will keep asking him for an acceptable student ID based on the available IDs in the database. Once he entered an acceptable stdID, the program will show the available degree(s) for this student (i.e. **BS**, **M**, **D**) in order to select from these degrees and store the selected degree(s). Then, the system will sleep for few seconds and it will redirect you to the menu window as in Figure 1.

2. **Menu Feature – def menuFeature**():

   It contains the setup of the menu window which contains eight features as illustrated in Figure 1. If you need to show the menu in your program, you need to call this function and assign number for each feature.

3. **Details Feature – def detailsFeature**():

   It shows the detail of the student like name, stdID, number of terms, college, and department. Here, you need to show this information on the screen, each record in one line as in Figure 2. Also, you need to store this information with the same style in **stdIDdetails.txt** file (i.e. std202006000details.txt) at same path of your jupyter file. After

showing and storing the records of the student, the program will clear all printed data on the screen, sleep for few seconds and redirect you to the menu window.

Name:
stdID:
Level(s):
Number of terms:
College(s):
Department(s):

Figure 2: Output of the details feature.

Here, you need to adjust the (level, number of terms, college, department) variables to show more than one value in same line based on the available levels and degrees for this student.

4. **Statistics Feature – def statisticsFeature():**

It shows some statistics about the student's records based on the selected option(s) by the user in the (**Start Feature**) and the results should follow the style in Figure 3.

```
========================================================
**************        Undergraduate Level        *************
========================================================
Overall average (major and minor) for all terms:
Average (major and minor) of each term:
        Term 1:
        Term 2:
        ...
        Term n:
Maximum grade(s) and in which term(s):
Minimum grade(s) and in which term(s):
Do you have any repeated course(s)?
========================================================
**************        Graduate(M) Level        *************
========================================================
Overall average (major and minor) for all terms:
Average (major and minor) of each term:
        Term 1:
        Term 2:
        ...
        Term n:
Maximum grade(s) and in which term(s):
Minimum grade(s) and in which term(s):
Do you have any repeated course(s)?
```

Figure 3: Output of the statistics feature.

Here, you need to show this information on the screen and you need to store them with the same style in **stdIDstatistics.txt** file (i.e. std202006000statistics.txt) at same path of your jupyter file. After showing and storing the records of the student, the program will clear all printed data on the screen, sleep for few seconds and redirect you to the menu window.

5. **Major Transcript Feature – def majorTranscriptFeature():**

It shows the transcript of the student based on the major courses and the selected option(s) as shown in Figure 4. This transcript shows the major courses, the average of major courses in each term and the overall major average for all terms up to the last term.

```
Name:                              stdID:
College:                           Department:
Major:                             Minor:
Level:                             Number of terms:
=======================================================
**************           Term 1           *************
=======================================================
course ID       course name        credit hours    grade
c1              course 1           3               80
c2              course 2           4               90
Major Average =                    Overall Average =
=======================================================
**************           Term 2           *************
=======================================================
course ID       course name        credit hours    grade
c3              course 3           3               95
c4              course 4           3               85
Major Average =                    Overall Average =
=======================================================
************** End of Transcript for Level (U)    *************
=======================================================
```

Figure 4: Major transcript.

Here, you need to show this information on the screen and you need to store them with the same style in **stdIDMajorTranscript.txt** file (i.e. std202006000MajorTranscript.txt) at same path of your jupyter file. After showing and storing the records of the student, the program will clear all printed data on the screen, sleep for few seconds and redirect you to the menu window.

6. **Minor Transcript Feature – def minorTranscriptFeature():**

It shows the transcript of the student based on the minor courses as shown in Figure 5. This transcript shows the minor courses, the average of minor courses in each term and the overall minor average for all terms up to the last term.

```
Name:                              stdID:
College:                           Department:
Major:                             Minor:
Level:                             Number of terms:
=======================================================
**************           Term 1           *************
=======================================================
course ID       course name        credit hours    grade
c6              course 6           3               80
c7              course 7           4               90
Minor Average =                    Overall Average =
=======================================================
**************           Term 2           *************
=======================================================
course ID       course name        credit hours    grade
c8              course 8           3               95
c9              course 9           3               85
Minor Average =                    Overall Average =
=======================================================
************** End of Transcript for Level (U)    *************
=======================================================
```

Figure 5: Minor transcript.

Here, you need to show this information on the screen and you need to store them in same style in **stdIDMinorTranscript.txt** file (i.e. std202006000MinorTranscript.txt) at same path of your jupyter file. After showing and storing the records of the student, the program will clear all printed data on the screen, sleep for few seconds and redirect you to the menu window.

7. **Full Transcript Feature – def fullTranscriptFeature():**

It shows the transcript of the student based on both minor and major courses as shown in Figure 6. This transcript shows the courses, the average of courses in each term and the overall average for all terms up to the last term.

```
Name:                           stdID:
College:                        Department:
Major:                          Minor:
Level:                          Number of terms:
=====================================================
**************          Term 1          **************
=====================================================
course ID      course name      credit hours    grade
c1             course 1         3               80
c2             course 2         4               90
c6             course 6         3               80
c7             course 7         4               90
Major Average =                 Minor Average =
Term Average =                  Overall Average =
=====================================================
**************          Term 2          **************
=====================================================
course ID      course name      credit hours    grade
c3             course 3         3               95
c4             course 4         3               85
c8             course 8         3               95
c9             course 9         3               85
Major Average =                 Minor Average =
Term Average =                  Overall Average =
=====================================================
************** End of Transcript for Level (U)   **************
=====================================================
```

Figure 6: Full transcript.

Here, you need to show this information on the screen and you need to store them in same style in **stdIDFullTranscript.txt** file (i.e. std202006000FullTranscript.txt) at same path of your jupyter file. After showing and storing the records of the student, the program will clear all printed data on the screen, sleep for few seconds and redirect you to the menu window.

Note that for features (5, 6, and 7), the student can select (BS, M and D) options, so, you need to show the transcripts for the selected levels on the screen and store them in same file staring with the lowest level to the highest level.

8. **Previous Requests Feature – def previousRequestsFeature():**

It shows the previous requests for this student. For example, student requested major transcript in previous term and full transcript in this term, the output should be shown as in Figure 7:

```
Request              Date              Time
=====================================================
Major                22/09/2020        13:30 PM
Full                 12/02/2021        14:30 PM
```

Figure 7: Previous requests.

Here, you need to show this information on the screen and you need to store them in same style in **stdIDPreviousRequests.txt** file (i.e. std202006000PreviousRequests.txt) at same path of your jupyter file. After showing and storing the transcript of the student, the program will clear all printed data on the screen, sleep for few seconds and redirect you to the menu window.

9. **New student Feature – def newStudentFeature():**

This feature allows the user to start the program for another student after clearing all previous data on the screen.

10. **Terminate Feature** – **def terminateFeature**():

This feature allows the user to terminate the program and at same time it will show the number of requests during the session at the screen.

1.2 Inputs/Outputs:

1. **Inputs:**

   • Input data are .csv files. You are going to read .csv files in Figure 8, in order to use them as inputs.
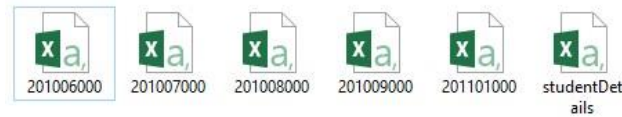

Figure 8: Input files.

   • As a hint, you can use "numpy" module to read .csv file by calling **numpy.loadtxt** and passing the suitable variables or you can use "pandas" module by calling **pandas.read_csv**.

   • Two main .csv files: first file is studentDetails.csv file which includes the students' IDs, names, colleges, and departments. The second file(s) is/are named by the student ID and it contains all student's records as in Figure 9 and Figure 10.

Serial	stdID	Name	College	Department	Level	Degree	Major	Minor	Terms
1	201006000	Ibrahim Ahmed Nemer	College1	C1Department1	U	BS1	C1D1MajorBS1	C1D1MinorBS1	8
2	201006000	Ibrahim Ahmed Nemer	College1	C1Department1	G	M1	C1D1MajorM1	C1D1MinorM1	4
3	201006000	Ibrahim Ahmed Nemer	College1	C1Department1	G	D1	C1D1MajorD1	C1D1MinorD1	6
4	201007000	Ali Mohamamd Hadi	College2	C2Department1	U	BS1	C2D1MajorBS1	C2D1MinorBS1	7
5	201008000	Ahmed Ali Mohammad	College3	C3Department1	G	M1	C3D1MajorM1	C3D1MinorM1	4
6	201008000	Ahmed Ali Mohammad	College1	C1Department1	G	M2	C1D1MajorM2	C1D1MinorM2	3
7	201009000	Mohammad Ibrahim Ali	College4	C4Department2	U	BS1	C4D2MajorBS1	C4D2MinorBS1	8
8	201009000	Mohammad Ibrahim Ali	College4	C4Department2	G	M1	C4D2MajorM1	C4D2MinorM1	3
9	201101000	Kamal Abdallah Mohammad	College5	C5Department1	G	D1	C5D1MajorD1	C5D1MinorD1	7

Figure 9: studentDetails.csv Input file.

Level	Degree	Term	courseName	courseID	courseType	creditHours	Grade
U	BS1	1	CourseA1	CA1	Major	3	75
U	BS1	1	CourseA6	CA6	Minor	4	76
U	BS1	1	CourseA2	CA2	Major	3	77
U	BS1	2	CourseA11	CA11	Major	3	78
U	BS1	2	CourseA16	CA16	Minor	4	60
U	BS1	2	CourseA12	CA12	Major	3	81
U	BS1	3	CourseA21	CA21	Major	3	82
U	BS1	3	CourseA26	CA26	Minor	4	81
U	BS1	3	CourseA22	CA22	Major	3	82
U	BS1	4	CourseA31	CA31	Major	3	83

Figure 10: one of the ID.csv Input file (i.e. 201006000.csv).

   • You can build a separate function for data generation and processing.

   • You are not allowed to modify the input files manually.

   • You need to decide your input variables for all functions, I left them empty.

2. **Outputs:**

   • You are going to show your outputs as described in the figures on the screen and you need to save same outputs in .txt files with same style as we discussed it before for each function.

   • You can build a separate function for transcripts generation process.

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).

   (f) Saving data at the end of the session to a file.

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) User input must be validated by the program i.e. valid range and valid type.

4. Students will not be forced to use object oriented paradigm.

5. Students are allowed to use the following modules **(datetime, numpy, pandas, random, time, sys, os, math, statistics, IPython.display)**. If you need to use other than these modules, you are required to get a permission first.

6. Your program must contain as many functions as needed. You need to divide your problem into small tasks and each task handled by a function as we explained in the description of the features.

7. The deadline for submitting the lab project is **end of week fourteen (Saturday December 11, 2021 before midnight)**.

8. 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 and you should use the template on the blackboard (LAB-ICS-104SecXX-GroupXX.ipynb).

   (b) The report as part of the Jupyter notebook (LAB-ICS-104-SecXX-GroupXX.ipynb). 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 5%)** (c) Your jupyter file should have the following lines at the beginning:

```
# This work done by group $$:
# Name of First Student, ID, Percentage of his work contribution.
# Name of Second Student, ID, Percentage of his contribution.
```
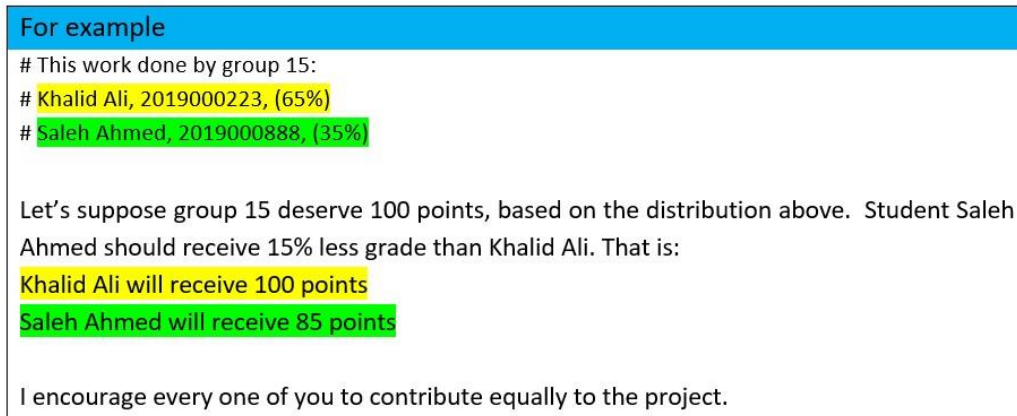
Figure 11: Start of your code.

> **For example**
> # This work done by group 15:
> # Khalid Ali, 2019000223, (65%)
> # Saleh Ahmed, 2019000888, (35%)
>
> Let's suppose group 15 deserve 100 points, based on the distribution above. Student Saleh Ahmed should receive 15% less grade than Khalid Ali. That is:
> Khalid Ali will receive 100 points
> Saleh Ahmed will receive 85 points
>
> I encourage every one of you to contribute equally to the project.

Figure 12: Grade distribution based on the student's contribution

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).

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

Note that, 20% of the grade are highlighted above and the remaining 80% will be on the code itself and the demo.