**GMI G e r m a n - M a l a y s i a n I n s t i t u t e**

**D I P L O M A P R O G R A M M E**

**A SSIG N M E N T 2**

**A c a de m i c P e r i o d: J an / J u l y 2 0 2 4**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Code** | | **SDT 2113** | | | **Course Name** | | **Basic of Software Engineering** | | | |
| **Title** | | **Simple Grading System** | | | **Examiner** | | Miss Norulmubarakah | | | |
| **Name** | | 1. Muhammad Haziq Bin Samsul Farid (SWE24070523) 2. Zikry Fahmi Bin Zamri (SWE24070260) 3. Arife Anuar Bin Muhammad Nor Hakim (SWE24070015) 4. Adyputera Ilmy Shah Bin Adrianizam Shah (SWE24070831) | | | **Program/ Group** | | **DSWE 5** | | | |
| **Reminder: GMI considers plagiarism and/or copying as a serious breach of academic conduct. Students who are caught doing so will face severe penalty.** | | | | | | | | | | |
| **No** | **Assessment Criteria** | | **CLO** | **PO** | **Level of Diff** | **Full Marks** | | **Score** | **Marks** | **Comment** |
|  |  | |  |  |  |  | |  |  |  |
|  |  | |  |  |  |  | |  |  |
|  |  | |  |  |  |  | |  |  |
|  |  | |  |  |  |  | |  |  |
|  |  | |  |  |  |  | |  |  |
|  |  | |  |  |  |  | |  |  |
|  |  | |  |  |  |  | |  |  |
|  |  | |  |  |  |  | |  |  |
|  |  | |  |  |  |  | |  |  |
|  |  | |  |  |  |  | |  |  |
|  |  | |  |  |  |  | |  |  |
|  |  | |  |  |  |  | |  |  |
|  |  | |  |  |  |  | |  |  |
| **Total** | | | | | | **100** | |  |  |  |
| **Total Marks** | | | | | |  | | | | |
| **Total** | | | | **PO** | |  | | | | |
| **PO** | |  | | | | |
|  | |  | | | | |
| **Signature** | | | | | |  | | | | |



ASSIGNMENT 2

**INTRODUCTION**

This project was made as a means to calculate a student’s grade, determining their letter grades, and make up a report of the result. While there are many more viable means that provide the same service, our grading system was made with simplicity in mind both in code writing and user experience.

Our grading system works with the user inserting their name and the grades of their subjects in the corresponding page shown from the menu. Then, all of the calculations will be made and their results displayed in other pages.

This grading system and its codes will be written in Python programming language. It is expected to be able to define and use functions, work with basic structures such as lists and dictionaries as well as implementing conditional statements and loops.

**PROBLEM STATEMENT**

We made this grading system with several reasons in mind. For some, we realize that many programs are run online thus need a stable connection to provide services. Some other programs are also too convoluted to work on. Next, users only need a specific result from the program.

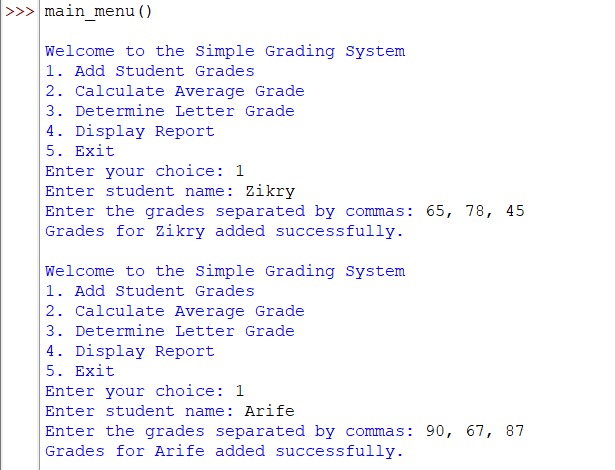
* Convoluted and complex grading system might leave the user clueless. Too many pages and required inputs could deter users from continuing in using the system and find a simpler method.
* Most other grading systems online might require a stable connection to provide a consistent service. Thus, users with no such connection would not be able to use the service smoothly or are unable to run it at all.
* Many other systems and programs require more information to work and provide results that user do not need. For example, user might only need to know their letter grades and average marks but need to provide more inputs than needed to fill in spaces required by the system.

**ANALYSIS AND DISCUSSION**

As an answer to the problems above, we made our grading system to adhere to the following standards:

* A Simple Interface: Our grading system are made to minimize as much pages and menus as possible to provide an ease of usage. This is to avoid turning our grading system into a maze of words.
* Able To Run Without A Connection: Our grading system will not be affected by such, as it only runs with the written codes in the program. Users with this program stored in their device would be able to run it anytime.
* Minimal Inputs From The User: We made our grading system to require as less inputs from the user as possible while maintaining the quality and accuracy of the service. Users are only required to enter their name, the number of subjects, and the grades for its subjects.

This is how our CLI will look like at the end :

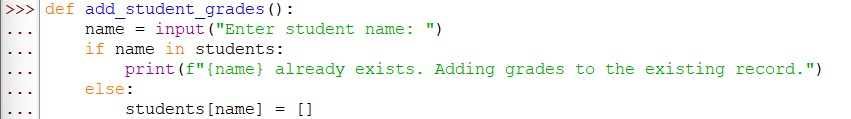


The following is a few parts of our codes as well as a brief explanation on how it works;

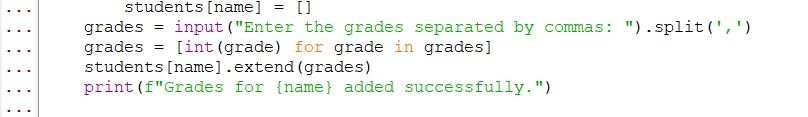
**STUDENT GRADES**

****

* This current dictionary contains no key-value pairs.
* This creates a dictionary to store student names as keys and their grades as values. Users can add as many student names as you want since the dictionary has no limit.

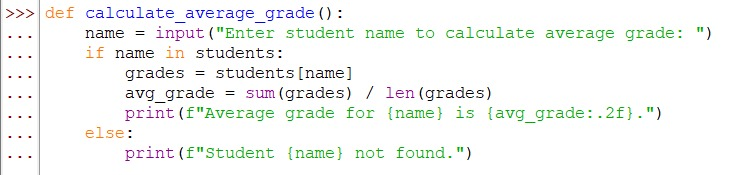
****

* The first line takes input from the user, asking them to enter the student's name, and stores it in the name variable.
  + ‘def’ = creates a function. In this case, the function named ‘add\_student\_grades’
  + ‘input()’ = functions that make the user enter a student's name as a text.
  + ‘if’ = checks a condition to decide what happened next.
  + ‘in’ = checks whether the value exists in the dictionary
* If the student already exists, it prints a message indicating that the student's record is being updated with additional grades. If the student's name doesn't exist in the student dictionary, the code is executed.
  + ‘f’ = an f-string, which allows users to insert variables directly into the string using curly braces {}.
  + ‘students[name] = []’ = create new line for the students dictionary.

****

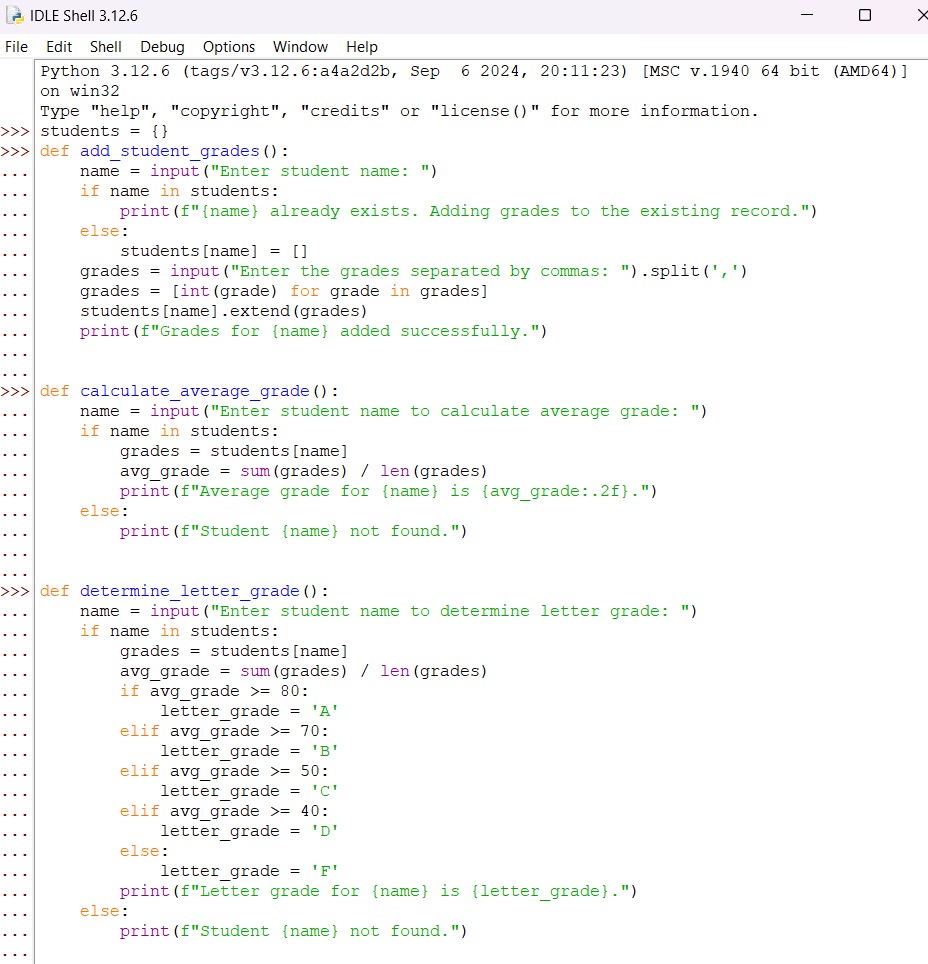
* Initializes a list for a particular student, identified by the variable name. This list will hold the student's grades.
* The program prompts the user to input grades for the student. The grades are separated by commas. It converts the list of grades into integers.
  + ‘split(',')’ = takes that string and breaks it into a list of separate grade values, splitting at the commas.
  + ‘int(grade)’ = converts each grade from a string into an integer.
  + ‘for’ = loop goes through every item in the grades list to convert each one.
  + ‘.extend()’ = takes a list that users enter and adds them to another list.
* The program prints a message confirming that the grades for the student have been successfully added, using an f-string to include the student's name dynamically.

**CALCULATE AVERAGE GRADE**

****

* Defines a function and the purpose of this function is to calculate and print the average grade for a student.
  + ‘def calculate\_average\_grade():’ = creates a new function named ‘calculate\_average\_grade’.
  + ‘grades = students[name]’ = This retrieves the list of grades for the student from the ‘students’ dictionary and stores it in the variable ‘grades’.
* Prints the average grade for the student, rounded to two decimal places. The f-string dynamically includes the student's name and the calculated average grade.
  + ‘sum(grades)’ = adds up all the grades in the list.
  + ‘len(grades)’ = gets the number of grades in the list
  + .2f = formats numbers into two decimal places
* Checks if the entered student's name exists in the students dictionary. If the name is found, the function proceeds to calculate the average grade; otherwise, it displays a message saying the student was not found..
  + ‘print(f"Student {name} not found."):’ = This block is executed if the student's name is not found in the students dictionary.

**DETERMINE LETTER GRADE**



* This function will calculate a student’s letter grade (A, B, C and so on) based on their average grade.
  + if avg\_grade >= 80

letter\_grade = ‘A’

= check the condition and assign the letter grade A if it passes

* + elif avg\_grade >=70

letter\_grade = ‘B’

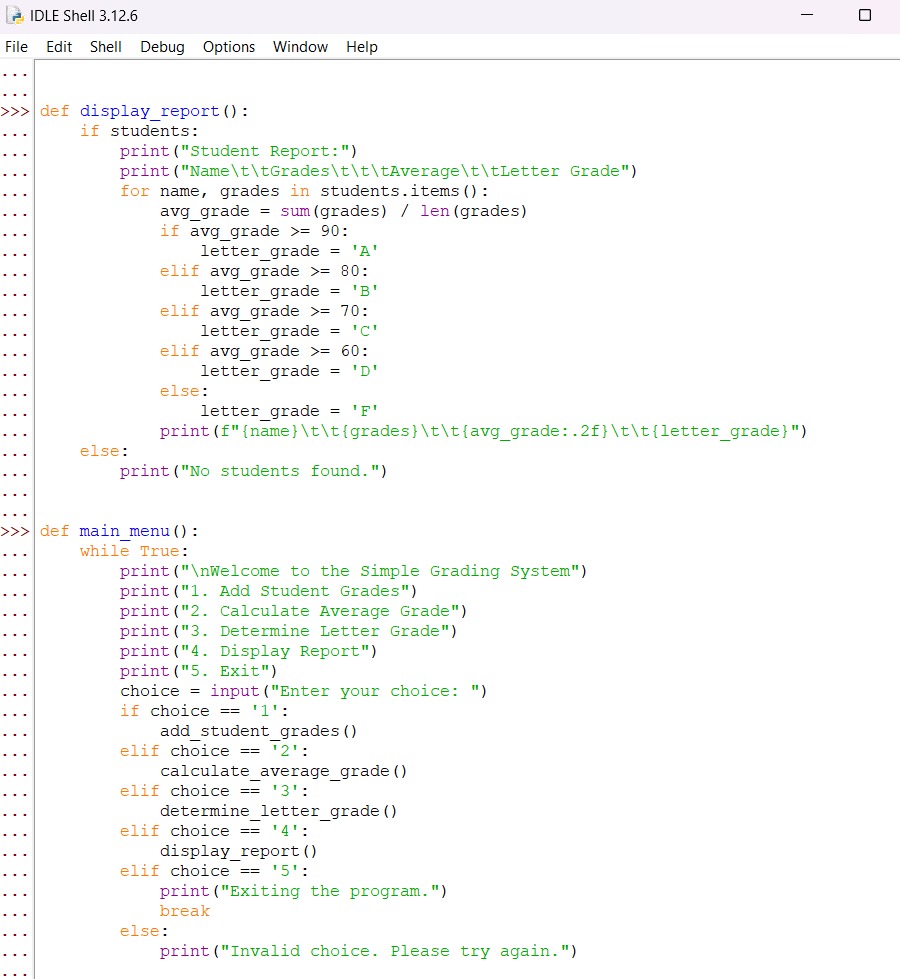
= check multiple conditions of avg\_ grade and assign the letter grade accordingly

* else = if any condition from ‘if’ and ‘elif’ wasn’t true, the code under ‘else’ will run.

if = check the conditions

elif = (else if) check multiple conditions

**DISPLAY REPORT**

****

* This function will display a summary report of the student’s grade, average mark and their letter grade
  + def display\_report() :

= creates a new dictionary called ‘display\_report’

* + for name, grades in students.items():

= a ‘for’ loop that goes through every student in the students dictionary and ‘students.items()’ returns each student’s name and their list of grades, which are stored in 'name' and 'grades'.

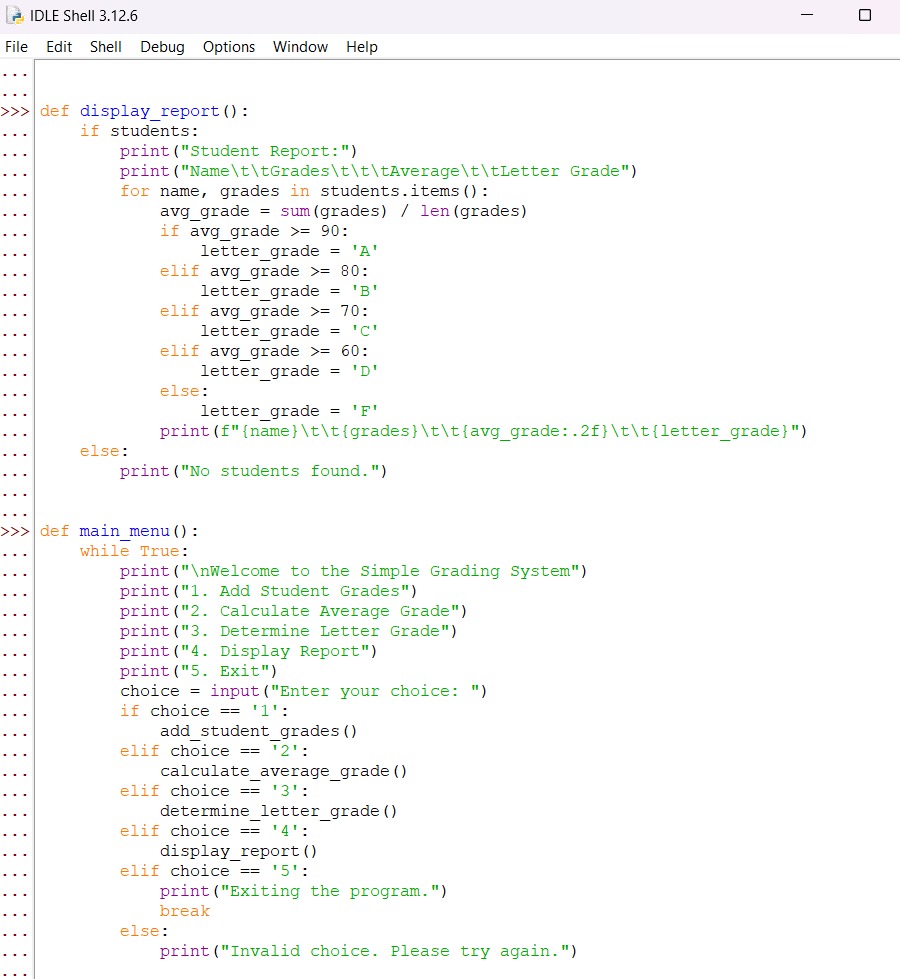
* + print(f"{name}\t\t{grades}\t\t{avg\_grade:.2f}\t\t{letter\_grade}")

f = indicates an f-string (allowing variables inserted directly into a string by placing them inside curly braces, {} )

\t = tab character, adds horizontal space (about 4-8 spaces) between texts

.2f = formats numbers (especially floats) into two decimal places

**MAIN MENU**



def main\_menu creates a function which will be labeled as “main\_menu”. This will be the main page that first greets the user.

* + while True:

= creates an infinite loop that keeps displaying the menu until the user decides to exit.

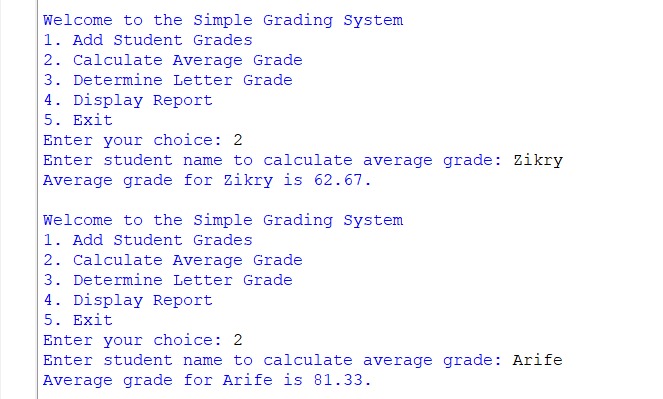
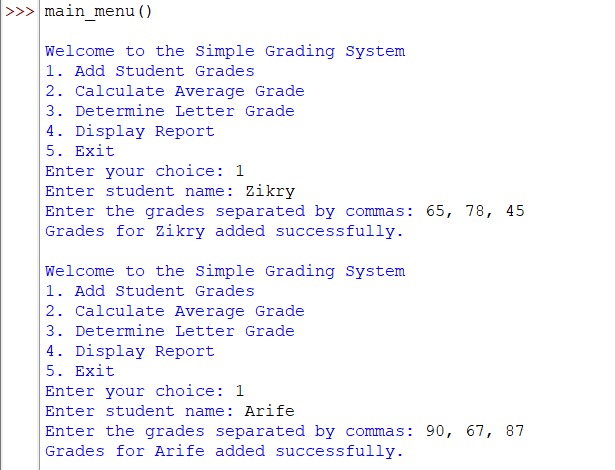
* + choice = input("Enter your choice: ")

= requires user’s input of which conditions would be checked to continue

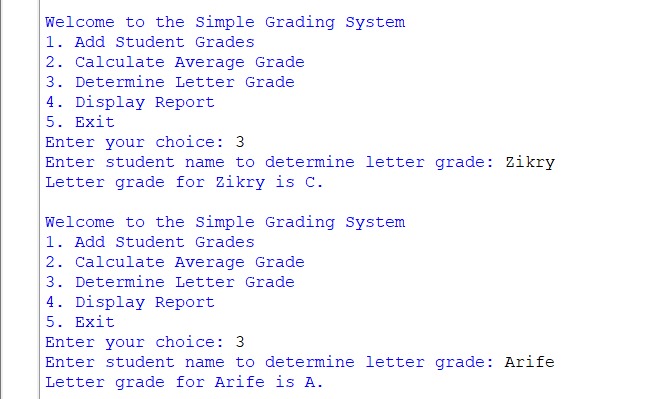
* + ‘= =’ is used to compare two values to check if they are equal. Using only one equal symbol ‘=’ here sould cause an error since it is used only to asssign something.
  + break

= This exits the loop and ends the program if the user chooses option 5 (Exit).

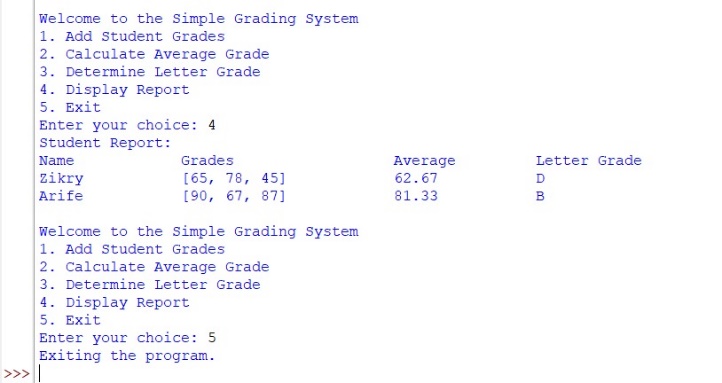
**Command-Line Interface (CLI)**



1. Add student Grades 2. Calculate average grade



3. Determine letter grades



4. Display Report

5. Exit

**CONCLUSION**

Our grading system might not be the most viable option in the field, but it could still provide the expected results of the users. But with the simplicity of the program and its codes, we hope that we could use it as a reference and guide to produce more programs at the side of providing a basic understanding of Python language and how to utilize it.

From this project alone, we as a team have had the opportunity to not only collaborate with each other into reaching the same goal, but to familiarize ourselves into the world of programming. We managed to obtain a grasp on the basics of the Python language, how it works and how to implement it into making a working program that is this grading system.