**Date: 2024-**

**Assignment 3**

This assignment consists of the programming questions related to the topics of week 1 and week 2. The main topics of questions are: Python IO and OOP.

All the students are required to follow the format of the program as specified in the guideline below.

1. All the programs should have initial **doc string** comment (‘’’ description of program‘’’) mentioning what your program will do.
2. Try to maintain single/multi-line comments in the place where needed to make the program understandable.
3. Maintain proper indention and newline spaces to increase the readability of the program.
4. The deliverable are 2 type of files (a single word file and multiple python program files):
   1. Separate python program files with **.py** extension (e.g. program\_name.py). Provide a relevant name to your program file on the basis of functionality of the program.
   2. A word file describing the working of all the programs according to their number. The details required in this is the description of program, screenshot of the testing (input given and output obtained in the execution environment such as IDLE or Command prompt or terminal whichever you prefer.). It is preferred that you work with multiple inputs and outputs.

**Questions**

1. Write a program to read a file “students.csv” and print it’s value. The fields to read are name, id, course, level and section.
2. Write a program to take the user details as the input from the user and write it to the existing file “students.csv”. The new record should be added to the end of the file. The fields to take input are name, id, course, level and section.
3. Write a program input any list of numbers from the user, calculate the addition, subtraction, multiplication and division values and write into a file with the current date and time. The program should allow the user to repeat the operation until they want. When the user choose the option to exit the program, the user should be displayed with the details in the current file in a proper format.
4. Write a program to take read a file and write it into another file. The input and output file names should be taken input from the user. Use the concept of try/except to handle the exception. Provide the proper error if the file does not exist while reading and also if the file for output exists.
5. Create a class Student with the attributes such as id, name, address, admission year, level, section. Instantiate the object of class to take input for all the attributes and display the output.
6. Write a program to create a class called recipe with the attributes such as id, name, ingredients, descriptions. Create another class called recipe book to manage the collection of recipes.
7. Write a program to implement a class called employee with attributes such as empid, name, address, contact\_number, spouse\_name, number\_of\_child, salary. Instantiate this class to input the values for multiple employees and write it in a file “employees.csv”. Allow the user of your program to see the list of employees and their details as well. Try to use the concept of try/except too in the program.
8. Write a program to implement a basic library book management with the functionalities such as issue the book, return the book and search the book. Use the concept of OOP to create the necessary classes on your own and implement the concept of other OOP features. For the storage of book details, use the file handling along with the exception handling.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***End of Assignment 3** \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*