Department of Information Technology – University of the Punjab
Programming for AI – MPhil/PhD (AI) F22
Lab-04

Max Time: 2.5 hours **Date: 11-1-2023**

Topics: Problem solving, operators, basic constructs, selection, repetition

Instructions:

- Please provide your own solutions and **DO NOT COPY** the code from your colleagues or the web.
- You can discuss your problems only with the teachers.
- All tasks carry equal points.

Task 1 [30]

Write a class named Patient that has attributes for the following data:

- First name, middle name, and last name
- Address, city, province, and ZIP code
- Phone number
- Name and phone number of emergency contact

The Patient class's _ _init_ _ method should accept an argument for each attribute. The Patient class should also have accessor and mutator methods for each attribute.

Next, write a class named Procedure that represents a medical procedure that has been performed on a patient. The Procedure class should have attributes for the following data:

- Name of the procedure
- Date of the procedure
- Name of the practitioner who performed the procedure
- Charges for the procedure

The Procedure class's _ _init_ _ method should accept an argument for each attribute. The Procedure class should also have accessor and mutator methods for each attribute.

Next, write a program that creates an instance of the Patient class, initialized with sample data. Then, create three instances of the Procedure class, initialized with the following data:

Procedure #1:	**Procedure #2:**	**Procedure #3:**
Procedure name: Physical Exam Date: Today's date Practitioner: Dr. Iram Charge: 2500.00	Procedure name: X-ray Date: Today's date Practitioner: Dr. Junaid Charge: 5000.00	Procedure name: Blood test Date: Today's date Practitioner: Dr. Suleman Charge: 2000.00

The program should display the patient's information, information about all three of the procedures, and the total charges of the three procedures.

Task 2 [20]

Write an Employee class that keeps data attributes for the following pieces of information:

- Employee name
- Employee number
- Department

Next, write a class named ProductionWorker that is a subclass of the Employee class. The ProductionWorker class should keep data attributes for the following information:

- Shift number (an integer, such as 1, 2, or 3)
- Hourly pay rate

The workday is divided into two shifts: day and night. The shift attribute will hold an integer value representing the shift that the employee works. The day shift is shift 1 and the night shift is shift 2. Write the appropriate accessor and mutator methods for each class.

Once you have written the classes, write a program that creates an object of the ProductionWorker class and prompts the user to enter data for each of the object's data attributes. Store the data in the object, then use the object's accessor methods to retrieve it and display it on the screen.

Task 3 [10]

In a particular factory, a shift supervisor is a salaried employee who supervises a shift. In addition to a salary, the shift supervisor earns a yearly bonus when his or her shift meets production goals. Write a ShiftSupervisor class that is a subclass of the Employee class you created in Task 2. The ShiftSupervisor class should keep a data attribute for the annual salary, and a data attribute for the annual production bonus that a shift supervisor has earned. Demonstrate the class by writing a program that uses a ShiftSupervisor object.

Note: The following task is **mandatory for PhD students.** However, MPhil students can perform this task as a bonus task.

Task 4 [10]

This exercise assumes you have created the Employee class for Task 2.
Create a program that stores Employee objects in a dictionary. Use the employee number number as the key. The program should present a menu that lets the user perform the following actions:

- Look up an employee in the dictionary
- Add a new employee to the dictionary
- Change an existing employee's name, and department, in the dictionary
- Delete an employee from the dictionary
- Quit the program

When the program ends, it should *pickle* the dictionary and save it to a file. Each time the program starts, it should try to load the *pickle*d dictionary from the file. If the file does not exist, the program should start with an empty dictionary.