22ITPC406 – Python Programming

UNIT – III (Assignment – 5)

Due by 15.04.2024

Instructions

- **Write the questions and answers.**
- **Run the programs and write the results for your data.**
- **Copying is strictly not acceptable**

1. Create a class such that it should get the price of the petrol per litre and the number of litres filled by each customer in one method. The other method should calculate the petrol bill and display it. The main program should allow the user to enter data for three customers and display it.

2. Create a class such that it should get the roll number, name, and marks of each student in one method. The other method should display the roll number, name, and marks of each student. Instantiate four student objects and call the methods in such a way to display the mark list in three columns.

3. Create a class such that it should get the name of the student in a method. Appropriate attributes must be used for counting the number of students and for the roll number. The roll number should be assigned to the student automatically when the student object is instantiated. Then it should display the serial number, name, and roll number of each student. Instantiate objects and display the serial number, name, and roll number of each student.

4. Create a Class (for points in two-dimensional space) that consists of a constructor and two methods.

   One method is to find the Euclidean distance between the origin (0,0) and the point in a two-dimensional space (x, y) as follows: $\sqrt{x^2 + y^2}$

   Another function is to find the Euclidean distance between the two points $(x_{1,} y_1)$ and $(x_{2,} y_2)$in a two-dimensional space as follows: $\sqrt{(\boldsymbol{x_1} - \boldsymbol{x_2})^2 + (\boldsymbol{y_1} - \boldsymbol{y_2})^2}$

   Create a two-dimensional point as the object of the class and compute the distance between origin and the point.

   Create two-dimensional points as the objects of the class and compute the distance between those two points.

5. (a) Make a class called User. Create two attributes called first_name and last_name, and then create several other attributes that are typically stored in a user profile. Make a method called describe_user() that prints a summary of the user's information. Make another method called greet_user() that prints a personalized greeting to the user.

Create several instances representing different users, and call both methods for each user.

(b) Add an attribute called login_attempts to your User class. Write a method called increment_login _attempts() that increments the value of login_attempts by 1. Write another method called reset_login_attempts() that resets the value of login_attempts to 0.

Make an instance of the User class and call increment_login_attempts() several times. Print the value of login_attempts to make sure it was incremented properly, and then call reset_login_attempts(). Print login_attempts again to make sure it was reset to 0.

6. Person is a base class constructed with name and age.

   Employee class is derived from Person class constructed with Designation and Salary.

   Write a method under employee class to show employee details: name, age, designation, and salary.

   Create objects of employee class and display employee details.

7. Create a Python program for the following:

   (i) Person is a base class constructed with name and age.

   (ii) Student is a base class constructed with id and room number.

   (iii) Resident is a class derived from person and student classes

   (iv) Write a method under resident class to show resident details: name, age, id, and room number.

   (v) Create objects of resident class and display the resident details.

8. Make a class called **Restaurant**. The **__init__() method** for Restaurant should store two attributes: a **restaurant_name** and a **cuisine_type**. Make a method called **describe_restaurant ()** that prints these two pieces of information, and a method called **open_restaurant()** that prints a message indicating that the restaurant is open.

   (a) Make an instance called **restaurant** from your class. Print the two attributes individually, and then call both methods.
   (b) **Ice Cream Stand:** An ice cream stand is a specific kind of restaurant. Write a class called **IceCreamStand** that inherits from the **Restaurant class** you created. Add an attribute called **flavors** that stores a list of ice cream flavors. Write a method that displays these flavors.
   Create an instance of IceCreamStand, and call this method.

9. Imagine a publishing company that markets both book and audiobook versions of its works.

Create a class **publication** that stores the **title** and price of a **publication**. From this class **derive two classes: book**, which adds a page count, **and tape**, which adds a playing time in minutes. Each of these three classes should have a **getdata()** method to get its data from the user at the keyboard, and a **putdata()** method to display its data.

Test the book and tape classes by creating instances of them, asking the user to fill in data with getdata(), and then displaying the data with putdata().

10. Create a class for distance. Distance is measured in terms of feet and inches.

    Write a special function to overload the binary operator ' > ' to compare two distances.

    Write a special function to overload the binary operator ' < ' to compare two distances.

    Write a special function to overload the binary operator ' + ' to add two distances.

    Write a method to display the distance.

    Get the inputs from the user in the form of feet and inches separated by space.

    Compare them using '<' and '>' operators and display appropriate messages.

    Add them and print the result using the display method.

11. Create a class for matrix. Write functions to
    (i) overload binary operator ' + ' to add two matrices.
    (ii) overload binary operator ' - ' to subtract them.
    (iii) overload binary operator ' * ' to multiply them.
    (iii) overload unary operator ‘-‘ to transpose a matrix.
    (iv) display the input matrices and the result matrices.

12. Create a class called *time* that has separate int member data for *hours, minutes*, and *seconds*.

    The constructor should initialize this data to 0. Use **__**str**__**() display it, in 11:59:59 format.

    (i) overload binary operator ' += ' to add two time objects and assign the result.
    (ii) overload binary operator ' < ' to compare and display the smallest time object.
    (iii) overload binary operator ' == ' to find whether they are equal.
    (iv) overload the operator ‘≥ ‘ to compare them.