Object-Oriented Programming Lab#7

Today's Topics

- Class and Object
- Constructor - Initialization of fields.
- Method Overloading
- Access Modifier and Encapsulation
- User Input - JOptionPane
- ArrayList

ArrayList:

Action	**Code**
Creating an ArrayList	ArrayList<T> list = new ArrayList<>();
Adding element to arraylist	list.add(new T());
Accessing an element	T t = list.get(int index)
Size of arraylist	int l = list.size();

An Employee Record System

Create an Employee Record System for the "UAP CSE" department. Implement the system in such a way where user can 1) add a new Employee to the system, 2) get the monthly salary of any employee, 3) increase the salary of an employee 4) promote an employee, 5) display the details of a specific employee, and 6 display the list of Employee and their info. Each Employee is identified by **his/her name, employee id, position/designation, rank, and salary.**

Object-Oriented Programming Lab#7

What you need to do:

1) Create an **Employee** class that has **4 private instance variables*; name, id, designation, salary***.

    a. Create a **constructor** that takes the initial value for those 4 attributes and initializes those attributes.

    Create the **following methods as** described

    a. public void increaseSalary(double amt)
    - Inside the method, increase the ***salary*** by ***amt*** amount.

    b. public void promote(String designation, double newSalary)
    - Inside the method, set the ***designation*** and ***salary*** to these 2 parameters.

    c. Getter method for all attributes

    i. public String getName()
    - This method returns the ***name***.

    ii. public String getId()
    - This method returns the ***id***.

    iii. public String getDsignation()
    - This method returns the ***designation***.

    iv. public double getSalary()
    - This method returns the ***salary***.

    d. public void display()
    - This method displays the attributes in the format "***Name:[name]; Id:[id]; Designation:[designation]; Salary:[salary]***".

2) Now create another class **UapCse** to represent the CSE department which has a list of Employees. So, there will be one attribute of type Employee **ArrayList** to represent the list of the employee [**ArrayList**<***Employee> employees***] and another attribute ***department*** to store the name of the department.

    a. Create a constructor and pass the department name as a parameter.
    - Inside the constructor initializes the ***department*** and instantiates the ***employees*** object.

    Add the following methods to this class.

    b. **private void addNewEmployee(Employee e)**
    - Add the **Employee** e to ***employees*** array.

    c. **public void addNewEmployee(String nm, String id, String des, double sal)**
    - Create an **Employee** object using the parameter provided and add the object to ***employees*** array by calling the **addNewEmployee(Employee e)** method.

    d. **private Employee findEmployee(String id)**

- Loop through the ***employees*** variable and find the **Employee** whose id matches with the parameter provided. If the Employee is found in the list, return the **Employee**. Return **null** otherwise.

e. **public void increaseSalary(String id, double amt)**
   - Find the Employee using the ***findEmployee(id)*** method. If the method returns an **Employee**, call the ***increaseSalary(double)*** method using that object.

f. **public void promote(String id, String newDesignation, double newSalary)**
   - Find the Employee using the ***findEmployee(id)*** method. If the method returns an **Employee**, call the ***promote(String, double)*** method using that object.

g. **double getSalary(String id)**
   - Find the Employee using the ***findEmployee(id)*** method. If the method returns an **Employee**, call the ***getSalary()*** method using that object.

h. **void display(String id)**

- Find the Employee using the ***findEmployee(id)*** method. If the method returns an **Employee**, call the ***display()*** method using that object.

i. **void display()**

- Loop through the ***employees*** and call display of Employee class for each item.

3) Create an **application class** (that has the main method) named "**Uap**" which will have the **main** method.

a. Inside the main method, create an object [name it ***myUap***] of **UapCse** class and then provide the following **menu** on the console. Once the user enters his/her option, you need to read the value and take appropriate action (See below) using the **myUap** object.

   - Input '1' to add a new Employee.
     If the user chooses this option, you have to ask the user for the employee's name, id, designation, and salary. After getting the value call **addEmployee(…)** method using **myUap** object.

   - Input '2' to get Salary info of a specific Employee
     If the user chooses this option, you have to ask the user for the employee id. After getting the id, call **getSalary(…)** method using **myUap** object and print the salary.

   - Input '3' to increase the salary of an Employee.
     If the user chooses this option, you have to ask the user for the employee id and the amount needs to be increased. After getting the input, call **increaseSalary(…)** method using **myUap** object.

   - Input '4' to promote an Employee.
     If the user chooses this option, you have to ask the user for the employee id, new designation and the new salary. After getting the input, call **promote(…)** method using **myUap** object.

- Input ‘5’ to display the details of a specific Employee.

  If the user chooses this option, you have to ask the user for the employee id. After getting the id, call **display(…)** method using **myUap** object and pass the id.

- Input ‘6’ to display the list of the Employees.

  If the user chooses this option, call **display()** method using **myUap** object.

- Input ‘0’ to exit the system.

  If the user chooses this option, exit the system.