DPIT121 – Lab Exercise 4

Due: Week 6 lab

You already submitted phase one of your project delivery as assignment I. In lab 4, you will continue on another iteration focusing on copy constructor, interface, comparable, Cloneable, shallow copy, deep copy, and sorting the list of objects.

Download and study lecture codes from week 4 (and all recorded lectures). In particular completely understand the advanced shallow and deep copy example in depth. Also, study Lab1 Solution from "Sample labs 4-6 and Solutions" folder which is similar to this lab.

1) Add a copy constructor to your MyDate, Car, InsurancePolicy, ThirdPartyPolicy, ComprehensivePolicy, User, Address, and InsuranceCompany (0.5 marks) ✓

2) Implement Cloneable interface (override clone ( ) method) for MyDate, Car, InsurancePolicy, User, Address, and InsuranceCompany. Note that ThirdPartyPolicy and ComprehensivePolicy don't need to implement Cloneable as it is the same as their parent InsurancePolicy. (0.5 marks) ✓

3) Write static methods in InsurancePolicy class to make shallow and deep copy of an ArrayList of policies. Use only the clone ( ). Copy constructor is not needed as it doesn't work with inheritance and polymorphism: (0.5 marks)

ArrayList< InsurancePolicy > shallowCopy(ArrayList< InsurancePolicy> policies) ✓

ArrayList< InsurancePolicy> deepCopy(ArrayList< InsurancePolicy> policies) ✓

4) Write static methods in User class to make shallow and deep copy of an ArrayList of users (The same as 3). (0.5 marks) ✓

5) Add a method ArrayList< InsurancePolicy> deepCopyPolicies() to the User to create a deep copy of user's policies. (0.25 marks) ✓

6) Add a method ArrayList< InsurancePolicy> shallowCopyPolicies () to the User to create a shallow copy of user's policies. (0.25 marks)

7) Add a method ArrayList<User> deepCopyUsers() to the InsuranceCompany to create a deep copy of all the users within the company. (0.25 marks) ✓

8) Add a method ArrayList<User> shallowCopyUsers() to the InsuranceCompany to create a shallow copy of all the users within the company. (0.25 marks) ✓

9) Implement Comparable interface for InsurancePolicy to compare InsurancePolicies based on the expiry date. Note: you need to make MyDate Comparable too. (0.25 marks) ✓

Ask output

10) Implement Comparable interface for User based on the city name. Note: you need to make address comparable too. (0.25 marks) ✓

11) Add compareTo1() to the User based on the total policy premium payments for the policies he owns. This can be replaced with the compareTo() if required. Note: assume a constant flatRate. (0.25 marks) ✓

12) Add a method ArrayList< InsurancePolicy> sortPoliciesByDate() to the User to sort all the user's policies based on the expiry date (using Comparable) and return it as an ArrayList. Note: You need to use Collections.sort(). (0.25 marks) ✓

13) Add a method ArrayList<User> sortUsers() to the InsuranceCompany to sort all the company's users based on the city name (or total premium payments if you manually change compareTo() and compareTo1() in User class and return it as an ArrayList. (0.25 marks)

14) Add this to your test code: (0.75 marks)

- For a user call deepCopyPolicies() and shallowCopyPolicies() to make deep and shallow copies of user's policies and store these lists in the test code. ✓

- change the user's city to "New York" and add a new policy to his polices. ✓

- sort the user's polices by calling sortPoliciesByDate(). ✓

- print the shallow and deep copy lists as well as user's policies and see how these three lists are compared. ✓

- For an insuranceCompany call deepCopyUsers() and shallowCopUsers() to make deep and shallow copies of company's users and store these lists in the test code. ✓

- add a new user to the company. ✓

- sort the company's users by calling sortUsers(). ✓

- print the shallow and deep copy lists as well as company's users and see how these three lists are compared. ✓

- make a clone of the insuranceCompany in your test code and test to show it is a deep copy. ✓