Lesson 8 – Lists Homework

Day – 1

Take the demo code of MyStringList.java. In that example several methods are implemented for the type of String. Now you have to modify the same code for the type of Person. For searching method use lastName as key input.

```
class Person {
	private String lastName;
	private String firstName;
	private int age;
	// ------------------------------------------------------------
	public Person(String last, String first, int a) { // constructor
		lastName = last;
		firstName = first;
		age = a;
	}
	// ------------------------------------------------------------
	public String getLast() // get last name
	{
		return lastName;
	}
	@Override
	public String toString() {
		return "Person [lastName=" + lastName + "\n FirstName=" + firstName +
		"\nAge=" + age + "]";
	}
} // end class Person
```

Problem – 2 – Sample question of Previous batch Common Program Test from FPP/MPP board.

Problem 2. [Polymorphism & Data Structure] In the `prob2` package of your workspace, you are given fully implemented classes `Staff` and `Teacher`. You will also find a class `Statistics` in the `prob2` package, with an unimplemented method `computeSumOfSalaries`. Your task is to implement `computeSumOfSalaries`.

The `main` method of the `Main` class must first combine the two input lists of `Staff` and `Teacher` objects into a single list (using the `combine` method provided). You may find the interface `EmployeeData` useful for this purpose; this interface is provided in the `prob2` package, but you will need to implement it yourself.

The combined list should be passed into `computeSumOfSalaries`, which must then *polymorphically* compute the sum of all the salaries of all `Staff` and `Teacher` objects

in this combined list, by calling each object's `getSalary()` method; it must then return this computed value.

Requirements for this problem.

(1) You must compute the sum of all salaries using polymorphism. (For instance, if you obtain the sum of all salaries by first computing the sums of the salaries in each list separately, you will receive no credit.)

(2) Your implementation of `computeSumOfSalaries` may not check types (using `instanceof` or `getClass()`).

(3) You must use parameterized lists (which means that specify the particular type inside<>), not "raw" lists. (Example: This is a parameterized list: `List<Duck> list`. This is a "raw" list: `List list`.)

(4) You must add a proper `List` type in your implementations in the `Main` and `Statistics` classes.

(5) You must implement the combine method provided in the Main class for the purpose of combining the Staff and Teacher lists.

(6) You are allowed to modify declarations of Staff and Teacher (to support inheritance or interface implementation), but you must not remove the final keyword from either of these class declarations.

(7) There must not be any compilation errors or runtime errors in the solution that you submit.