Problem 1. Work with following lambda expression:

```
Random r = new Random();
 () ->  r.nextInt();
```

```
public class Problem1 {

        // name and type of lambda goes here




        // representing lambda as a method reference
        // Hint: To define the method reference, make use of a helper method.









        //representing lambda as a static nested class









        public void evaluator() {









        }

        public static void main(String[] args) {

                Problem1 p = new Problem1();

                p.evaluator();

        }

}
```

Problem 2. Use Lambdas and Streams to do the followings:

1) Use the Collectors.joining method to print out All Employee objects and separate each one with a delimiter of "---\n---".
2) Print a list of both the capital and non-capital last names of employees using stream concat operation.
3) Calculate sum of Employee salaries using DoubleStream.
4) Calculate sum of Employee salaries with Stream's reduce method.
5) Count number of Employees in each department by using groupingBy operation.
6) Print out each department name with the average salary by using groupingBy operation.
7) Print out each department and its corresponding employees by using groupingBy operation.
8) From given employee list, create Map<String, List<Double>> map: keys will be department names, and values will be salaries of the department by using groupingBy operation and show the result using forEach method.