**Problem I (10 Points):**

The Library is having a contest. Library members submit their ids so they can participate. The library will take a special list of books and go through the LibraryMember participants and, one by one, checkout the next available book on behalf of the next member in the participant list. If, during the process, a member is found who, in this checkout process, has just checked out his 10th book (that is, he now has exactly 10 CheckoutRecordEntries in his CheckoutRecord), he wins the contest.  
  
The code in prob1.Problem1 checks a list of LibraryMembers for a possible winner of the contest, using a List of books obtained from TestData. However, the code does not compile because the checkout method on LibaryMember is capable of throwing a LibrarySystemException and so this checked exception needs to be handled in the middle of a call to the current stream’s filter method.   
  
Use one of the standard techniques mentioned in class to fix this exception-handling problem so that the code compiles properly.

**Problem II (10 Points):**

Your package prob2.exam contains three subpackages, partA, partB, partC. Each contains a designated class file (respectively PartA, PartB, PartC), along with (possibly) other classes. You will pick **TWO** of these to work on – do not attempt all three (one of the class files should remain unchanged.)

At the top of each of the designated class files, you will see a lambda expression. There are several things you will need to do with this expression.

a. Assign an appropriate type (some functional interface)

b. Express it as a method expression

c. State the type of method expression you have used

d. Express it as an inner class

e. Evaluate the lambda, the method expression and the inner class inside an evaluator() method.

There is a main method in each of the designated class files that attempts to run the evaluator method. In the body of the evaluator method, you should test your typed lambda expression, your method reference, and your inner class operation.  
  
Each designated class provides a template for your work. You must follow this template. A sample solution is provided in the package prob2.sample. Follow the format of this sample very closely.

The lambdas provided in the three parts are:

PartA: () -> Math.random()

PartB: (CheckoutRecord record) -> record.getCheckoutEntries()

PartC:(Long a, Long b) -> a.compareTo(b)

NOTE: In partC, you may not type the lambda as a BiFunction.

**Problem III(10 Points):**

Reduce. In your package prob3.exam there are three classes: Person, FindOldestPerson, TestCode. The FindOldestPerson class contains an unimplemented static method findOldestPerson(List<Person> list). For this problem implement this method, which must return the oldest Person object occurring in the input list. Your implementation must use the reduce method for Streams. (If you do not use the reduce method, you will receive less than 50% credit for this problem.)   
  
The class TestCode provides test data to test your method. You may use this class, but do not modify it in any way.

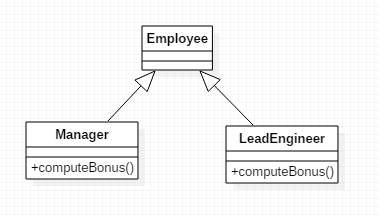
**Problem IV(10 Points):**

Exception-handling within lambda pipelines. High school students at Fairfield High School participate in an annual word contest. Students are given 15 minutes to type into a test computer as many words as they can think of that do not begin with one of the illegal letters. In this year’s contest, the illegal letters are A, B, C, E, M, N, R, S, T.   
  
The evaluator program for student submissions runs a method adjustWords which takes the list of words created by a participant, checks that all the words are legal, turns them into lower case words, and produces a new list with these modified words. The code for this is shown in the class WordGame in your prob4.exam package. However, it has been commented out because it is implemented as a lambda pipeline in which one of the lambda implementations throws an exception which is not supported by the interface type of the lambda.

Modify the implementation of this method so that the exception is handled in one of the standard ways. Then make sure that the main method executes test data as expected (you will need to uncomment the code in the main method).

**Problem V(10 Points):**

In your prob5 package, you will find three classes: Employee, Manager, LeadEngineer. Their relationships to each other are shown in the following class diagram:



The computeBonus methods in the classes Manager and LeadEngineer have identical implementations in both classes. Without changing the inheritance relationships shown here, use new features in Java 8 to modify the implementation of these classes so that there is no duplication of code.

**Problem VI (SCI 3 Points):**

Write one or two paragraphs (or more) relating the topics of the second half of the course to principles of SCI.