Lab8 Solutions

Problem 1

Short answer

a. For each lambda expression below, name the parameters and the free variables.

i.
```
Runnable r=()→ {

int[][] products = new int[s][t]; for (int i = 0; i < s;
i++) {

                    for(int j = i + 1; j < t; j++) {
                        products[i][j] = i * j;

} }

}
```

Parameters	Free Variables
i, j	s, t

ii.
```
Comparator<String> comp = (s, t) → {

             if(ignoreCase == true) {
                return s.compareToIgnoreCase(t);

} else{
return s.compareTo(t);

} }
```

Parameters	Free Variables
s, t	ignoreCase

b. An example of a method reference is:

`Math::random`
Its corresponding functional interface is `Supplier<Double>`. Do the following:

i. Rewrite this method reference as a lambda expression
ii. Put this method expression in a main method in a Java class and use it to print a random number to the console
iii. Create an equivalent Java class in which the functional behavior of `Math::random` is

expressed using an inner class (implementing `Supplier`); call this inner class from a main method and use it to output a random number to the console. The behavior should be the same as in part ii.

Coding Solution:
*Attached in eclipse file name **'MPP-Lab8'*** inside package **`'lab8.question_1b'`**

Problem 2.A

Look at the code in the package lesson8.lecture.comparator2. Suppose we sort using the sort method in the EmployeeInfo class together with the NameComparator. Look at the compare method in the NameComparator: If two Employee objects have the same name, what is the return value of compare? This tells us that these Employee objects should be *equal*, but is this always true? Give an example of two Employee objects having the same name but that should *not* be considered equal. Rewrite the compare method so that, if compare does return 0, the Employee objects are indeed equal. (This issue is known as *consistency with equals.*)

Coding Solution:
Attached in eclipse file name ***'MPP-Lab8'*** inside package
`'lab8.question_2a_comparator'`

Problem 2.B

Fix the `compare` method, as in part A, for the Comparator used in `lesson8.lecture.comparator3`

Coding Solution:
Attached in eclipse file name ***'MPP-Lab8'*** inside package
`'lab8.question_2b_comparator'`

Problem 2.C

Fix the `compare` method, as in part A, for the lambda expression used to `compare Employee` objects in `lesson8.lecture.lambdaexamples.comparator3`

Coding Solution:
Attached in eclipse file name ***'MPP-Lab8'*** inside package
`'lab8.question_2c_lambdacomparator'`

Problem 3

Consider the following lambda expression. Can this expression be correctly typed as a `BiFunction`? (See `lesson8.lecture.lambdaexamples.bifunction`.) (Hint: Yes it can.)

```
(x,y) -> { };
```

```
List<Double> list = new ArrayList<>(); list.add(Math.pow(x,y));
list.add(x * y);
return list;
```

Demonstrate you are right by doing the following: In the main method of a Java class, assign this lambda expression to an appropriate `BiFunction` and call the `apply` method with arguments `(2.0, 3.0)`, and print the result to console.

Coding Solution:
*Attached in eclipse file name **'MPP-Lab8'*** inside package '`lab8.question_3`'

Problem 4

Implement a method with the following signature and return type:

```
    public int countWords(List<String> words, char c, char d,
int len)
```

which counts the number of words in the input list `words` that have length equal to `len`, that contain the character `c`, and that do not contain the character `d`. Create a solution like the "Good" solution in `lesson8.lecture.filter` – a Good solution creates a lambda expression each time values are passed into countWords.

Coding Solution:
*Attached in eclipse file name **'MPP-Lab8'*** inside package '`lab8.question_4`'

Problem 5

Redo `lesson7.labs.prob3` in two different ways:

a. Use a lambda expression instead of directly defining a Consumer

b. Use a method reference in place of your lambda expression in (a)

Coding Solution:
*Attached in eclipse file name **'MPP-Lab8'*** inside package '`lab8.question_5`'

Problem 6

Finish the Examples exercise that was given in class (file: *Lambda and Method Reference Exercises*)

Coding Solution:
*Attached in eclipse file name **'MPP-Lab8'*** inside package '`lab8.question_6`'