


|   |   |
|---|---|
|  | <p><b>Computer Science Department/College of Engineering and Computer Science</b></p> <p><b>CSc 20: Programming Concepts and Methodology II</b></p> <p><b>Final project – Fall 2016</b></p> |
|---|---|

### Objective:

This final project is to put together all the key concepts which we learned so far in the class.

### Overview:

The final CSC 20 project is to keep records and perform analysis for a CSC 20 class of students. Our Fall 2016 class may have up to 17 students (for example). There are five labs (assumed we have five) during the section. Each student is identified by a four-digit CSUS student ID number.

The program is to output the student scores and calculate and print the statistics for each lab. The output is in the same order as the input; no sorting is needed. The input is to be read from a text file. The output from the program should be similar to the following:

Here is some sample data (for illustration only):

```
Stud L1 L2 L3 L4 L5
1234 78 83 87 91 86
2134 67 77 84 82 79
1852 77 89 93 87 71
```

```
High Score 78 89 93 91 86
Low Score 67 77 84 82 71
Average 73.4 83.0 88.2 86.6 78.6
```

Use one and two-dimensional arrays only. Test your program with the following data - Program should print all the lowest or highest scores for each lab.

Here is copy of **actual data** to be used for input.

```
Stud Lb1 Lb2 Lb3 Lb4 Lb5
1234 032 017 020 028 034
2134 030 036 030 017 030
3124 030 035 020 030 030
4532 031 017 031 032 027
5678 040 012 035 028 034
6134 034 040 035 018 025
7874 030 030 036 038 018
8026 040 010 026 028 016
9893 024 009 017 027 020
1947 025 020 028 023 035
```

2877 035 030 019 022 030  
3189 022 030 020 018 017  
4602 039 040 021 038 016  
5405 011 011 020 021 010  
6999 022 028 029 011 020

These Concepts MAY apply to this project;

1. Object Oriented Programming.
2. File IO.
3. Wrapper Classes.

Essentially you have to do the following:

- (1) Read Student data from a formatted file.
- (2) Compute High, Low and Average for each lab score.
- (3) Print the student data and statistical information.

### **Designing:**

This program can be written in one class. However, understanding division of responsibilities in each entity and linking them is at the heart of Object Oriented Design. Observe the classes you are being asked to create and analyze what you have learned from this design.

Keep in mind for following:

A clear demonstration through your implementation that you understand why Abstract classes and Interfaces exist.

Finding opportunities to use Abstract Classes and Interfaces.

Put each class in its own .java file.

**Code Snippets: The following code is only one of the design. There are other design options. Assume you declared:**

```
final int NUMBER_OF_CSC20_LABS = 5;

class Student
{
    private int SID;
    private int scores[] = new int[NUMBER_OF_CSC20_LABS];
    // write public getter and setter methods for
    // SID and scores
    // add methods to print values of instance variables.
}

class Statistics
{
    private int [] lowscores = new int [NUMBER_OF_CSC20_LABS];
```

```

private int [] highscores = new int [NUMBER_OF_CSC20_LABS];
private float [] avgscores = new float [NUMBER_OF_CSC20_LABS];

void calculateLow(Student [] a)
{
    // This method will find lowest score and store it in an array names lowscores
}

void calculateHigh(Student [] a)
{
    // This method will find highest score and store it in an array names highscores
}

void calculateAvg(Student [] a)
{
    // This method will find avg score for each lab and store it in an array named avgscores
}

// add methods to print values of instance variables.
}

class Util
{
    static Student [] readFile(String filename, Student [] stu)
    {
        // Reads the file and builds student array.
        // Open the file using FileReader Object.
        // In a loop read a line using readLine method.
        // Tokenize each line using StringTokenizer Object
        // Each token is converted from String to Integer using parseInt method
        // Value is then saved in the right property of Student Object.
    }
}

```

Putting it all together:

```

public static void main(String [] args)
{
    Student studArr[] = new Student[35];

    // populate the student array
    studArr = Util.readFile("filename.txt", studArr);
}

```

```

Statistics stat = new Statistics();
stat.calculateLow(studArr);

// add calls for the high and average values
// Print the data and statistics
}

```

### **Additional materials:**

Working with Text files:

// ReadSource.java shows how to work with readLine and FileReader

```

public class ReadSource
{
    public static void main(String[] arguments)
    {
        try
        {
            FileReader file = new FileReader("ReadSource.java");
            BufferedReader buff = new BufferedReader(file); String line;

            line = buff.readLine(); while (line != null)
            {
                System.out.println(line); line = buff.readLine();
            }
            buff.close();

        }
        catch (IOException e)
        {
            System.out.println("Error " + e.toString());
        }
    }
}

```

How do you tokenize a String?

The following example illustrates how the String.split method can be used to break up a string into its basic tokens:

```

String[] result = "this is a test".split("\\s");
for (int x=0; x<result.length; x++)
    System.out.println(result[x]);

```

How to convert a String to an Integer:

```
int x = Integer.parseInt(str);
```

**Submitting your work:**

- (1) Show the program's execution to your instructor.
- (2) Please turn in your work via SacCT along with your Java source programs, along with the pre/post conditions for each method in your classes, and its output (in MS Doc or PDF format) using actual data showed above.