SYSC 2004
Fall 2018
Lab 1

Blue text – Vocabulary terms that you must know and use in all assignments and exams.

Red text – Lessons I want to pay attention to, appreciate; or warnings.

`Courier Font – names of Java classes, objects, variables and methods. Uppercase letters and no-spaces matter!`

Objectives

1. Complete a program template to use the services of an existing class (write client code).

Part A

You are provided with two files – your first Java program with more than one file!

- Rectangle.java – A Java class that has been written for you and is ready to be used by you. It is a utility class or a service class.
- Lab1.java - Another program template, intended to help you learn in stages, with a little help. This is the client, with the main() method where your program will start execution. It is this file that you need to edit.

1. You now need TWO files in your project.
2. Use copy-and-paste to work efficiently.
3. Tasks 1, 2, 3 will exercise the calling of methods on an object – get used to this syntax. Type it over and over again to gain muscle memory.
4. Task 4 and 5 are tasks that you will likely have done in previous courses – finding the maximum and minimum in an array. Remember to draw on that experience first and then simple extend it to include those method calls on objects.

Part B

You will now add onto the code from **Part A** to do some more experiments with using some interesting existing classes.

- For each of the following tasks, simply add more code at the bottom of your main() method from Part A, and re-run the whole program.
- For each new class used, you are required to do a web search (Example: Java API String) and study the JavaDoc for the class to find the relevant methods to use.

1. The `Random` class provides a utility for generating random numbers. There is an example of its use in the provided code for **Part A** that you're already run– look at the code for **Part A** and find where it is used.
    a. Declare-and-construct a new `Random` object called `myRandomGenerator`
        - Use the 1-argument constructor for Random, using your own birthdate as the seed value = month * day * year
        - Example: If you were born on January 3, 1991, your seed should be 1*3*1991
    b. Generate 2000 random integers between the range of 65 and 90 (inclusive)
        - You will need to use the overloaded version of `nextInt()` that limits values between 0 and n-1
        - You still have a mathematical problem to solve to shift those values to between 65 and 90.
    c. Print out the minimum value, the maximum value and the average value (all as integers). Before continuing to the next exercise, confirm that you indeed generate numbers only in the given range; if not, you will have a bug in the next exercise.

2. The `String` class has already been introduced in class. Now you will use it in conjunction with the `Random` class.
    a. Declare a `int`-variable called `stringLength`;
    b. Initialize `stringLength` to a random value between 1 and 100
    c. Copy-paste the following code to declare an array of bytes, called `myBytes`

```
byte myBytes[] = new byte[stringLength];
```

    d. Using a loop, initialize each element in `myBytes` to a value between 65 and 90 (i.e. using the code that you wrote in the first exercise)
    e. Declare-and-construct a String object called `s1` using `myBytes` are its initial value
        - You should study the API of `String` to find a useful version of the constructor that will do this job.
    f. Print out the string `s1`
        - You've just written an automatic password generator!

g. Print out the lowercase version of `s1`
    - You should study the API of `String` to find a useful instance method that will do this job.

3. The `Date` class represents an instant in time, with millisecond precision. As a final quick exercise, print out the current time.
    a. Declare-and-construct a `Date` object called today, that contains the current time.
        - You should study the API of `Date` to find a useful constructor that will do this job.
        - You will have to add an import statement: `import.util.Date;`
    b. Print out the current time.