**Chapter 6**

Ex. 6.1 – Done

Ex 6.2 - Class documentation’s structure seems to have a set order of how things are introduced. First, we get a detailed description of the classes intended purpose and how to use it. Then we are given a summary of the classes fields (String only has one). Then it introduces all of the classes constructors with description about each one. Finally there is the method summary which is very long. It contains every method in the class and explains in detail how each one operates and is used.

Ex 6.3 – There are two versions of startsWith method in the String class. They both search a string to see if it starts with a certain “prefix”. The first only has one parameter and searches from the start of the string and returns true or false depending on if prefix was the same as the beginning of the string. The second version takes a second integer parameter and it uses to offset where it begins the search.

Ex. 6.4 – There is a method that test to see if a string ends with a given suffix. It was creatively named endsWith and has only one parameter, “suffix”, and it compares that to the end of the string instead of the beginning. Other than that, it functions the same a startsWith.

Ex. 6.5 – There is a method called length that returns the number of characters in a string. It has no parameter and returns an integer value.

Ex. 6.6 – To find the methods above I used control F and searched the page for what I thought would be key words in the method. It worked very well for finding endsWith and the word length was so numerous it took some scrolling to find the actual method.

Ex. 6.7 – public string trim()

String trimText = text.trim();

Ex. 6.8 – Implement an improvement to tech-support1 that make the method tolerant of an extra space around the word “bye”.

Done and saved.

Ex. 6.9 – Make it so that the word “bye” is no longer case sensitive.

Done and saved.

Ex. 6.10 – the equals method returns a Boolean.

Ex. 6.11 – Change your implementation to use the equals method instead of startsWith

Done and saved.

Ex. 6.12 – The random class is part of the java.util package and it is used to generate random numbers. To create an instance of Random we can use constructor calls. From what I am reading the random number is generated by a given “seed” which passes through a sequence of method calls to generate a “random” number.

Ex. 6.13– Random random= new Random()

int x = random;

Ex. 6.14– Write dome code to test the generation of random numbers. Create two methods (printOneRandom and printMultiRandom(int howMany).

Done and Saved.

Ex. 6.15– when 100 is used in nextInt method the possible random numbers generated will be between 0 and 99. This is because the parameter is exclusive.

Ex. 6.16 – Write a method in RandomTester classs called throwDie and have it return a value between 1 and 6.

Done and Saved.

Ex. 6.17 – Write a method called getResponse that randomly returns one of the strings “yes”, “no”, “maybe”. I wasn’t sure what class to put this in so it is currently in the Responder class.

Done and Saved.

Ex. 6.18 – Extend getResponse method so that it uses an ArrayList to store an arbitray number of responses and randomly returns one of them.

Done and Saved.

Ex. 6.19– Add a method to your RandomTester class that takes a parameter max and generates a random number between 1 and max(inclusive).

Done and saved.

Ex. 6.20– Add a method to RandomTester that takes two parameters (min & max) inclusive.

Done and saved.

Ex. 6.21– Yes the SecureRandom class could be used instead of the Random Class, other than being more secure it fills the same role. Security of random numbers can be incredibly important. Like when you get a verification code for altering accounts for various things. It can also be used to protect passwords (encryption) and pin numbers.

Ex. 6.22–

Ex. 6.23–

Ex. 6.24–

Ex. 6.25–

Ex. 6.26–

Ex. 6.27–

Ex. 6.28–

Ex. 6.29–

Ex. 6.30–

Ex. 6.31–

Ex. 6.32–

Ex. 6.33–

Ex. 6.34–

Ex. 6.35–

Ex. 6.36 –

Ex. 6.37 –

Ex. 6.38 –

Ex. 6.39 –

Ex. 6.40 –

Ex. 6.41 –

Ex. 6.42 –

Ex. 6.43 –

Ex. 6.44 –

Ex. 6.45 –

Ex. 6.46 –

Ex. 6.47 –

Ex. 6.48 –

Ex. 6.49 –

Ex. 6.50 –

Ex. 6.51 –

Ex. 6.52 –

Ex. 6.53 –

Ex. 6.54 –

Ex. 6.55 –

Ex. 6.56 –

Ex. 6.57 –

Ex. 6.58 –

Ex. 6.59 –

Ex. 6.60

Ex. 6.61

Ex. 6.62

Ex. 6.63

Ex. 6.64

Ex. 6.65

Ex. 6.66

Ex. 6.67

Ex. 6.68

Ex. 6.69

Ex. 6.70

Ex. 6.71

Ex. 6.72

Ex. 6.73

Ex. 6.74

Ex. 6.75

Ex. 6.76

Ex. 6.77

Ex. 6.78

Ex. 6.79

Ex. 6.80

Ex. 6.81

Ex. 6.82

Ex. 6.83

Ex. 6.84

Ex. 6.85

Ex. 6.86

Ex. 6.87

Ex. 6.88

Ex. 6.89

Ex. 6.90