**Computer Science 112 Spring 2013**

Lab 7

**Due by in class one week from today.**

Put the answers to the questions 1 and 2 in separate block comments at the top of the java file for part 3. This java file should be zipped together and submitted on moodle under “assignment 7.”

 1. True or False

* 1. The following is an example of a call to a method

 String s = “this”;

 boolean t = s.equals(“that”);

* 1. It is valid to call a method as follows

 int z = add(int x, int y);

* 1. Any line of code calling a method will contain parentheses
	2. The following compiles, even if x and y are already defined in main.

 public static int sum (int x, int y){return x+y;}

 public static double average (int x, int y){

 return (double)sum(x,y)/2;

 }

* 1. It is allowable to have a method call inside a println statement, e.g.

System.out.println(“The result: ” + sum(a,b));

* 1. It is possible for methods to return multiple values
	2. A class containing both of the following methods will throw an error:

 public static void printSum(int x, int y){...}

 public static void printSum(double x, double y){...}

* 1. The following compiles

 public static void main (String[] args){

 IntegerPrinter(3);

 public static void IntegerPrinter(int n){

 System.out.println(n);

 }

 }

1. Short Completion: write lines of code to accomplish each task.
	1. Declare a method which takes as input a String, a double, and an int, and which returns a boolean. The method doesn't have to do anything with the parameters.
	2. Write a method pn which takes an integer N and prints the numbers 1,2,...,N
	3. Write a method isSquare(int x) which determines if the input is a square or not, i.e. whether or not there is some integer y such that y\*y=x. Do this using a loop. Your program should print “yes x is a square” or “no x is not a square”
	4. Write a method max(int x, int y) which returns the integer which is larger.
	5. Write a method StringEqual which takes as input two strings and determines if they are equal. You may use the charAt and length methods of String but not the equals method. Your method should print something like “yes S and T are equal”
	6. Write a method StringSub which takes as input two strings S and T, and which determines if S appears as a substring of T. You may use the substring and equals methods of String. Your method should print something like “yes S appears as a substring of T”
	7. Write a method AveOfFile which takes as input a filename (which you may assume consists of lines of numbers) and returns the average of all numbers in that file.
2. Write a method removePunctuation which takes a String, removes all punctuation within that String, and returns the resulting String. For the purposes of this exercise, assume punctuation means a symbol from the following list:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| , | . | ; | : | ? | ! | ) | / | " | ( |

Write a method wordify which takes as input two filenames, which reads from the first file, and which writes to the second file. For each line in the first file, the method should loop through the line (similar to part 3 of lab6) and split it up into words which are separated by spaces. Each word should be written to the second file in such a way that in the end the second file contains only one word per line and should contain no punctuation. Your method should make use of removePunctuation by calling it, not by copying and pasting code from it.

Write a method lineify which takes as input two filenames, reads from the first file, and writes to the second file. In the end the second file should contain only one line, which has all the test from the first file. Spaces should be inserted so words don't get concatenated.

Write a method called countThe which takes a filename and returns as an integer the number of times the word the appears in the file (uppercase or lowercase). Write a method called countWords which takes a filename and returns as an integer the number of words in the file. Both methods should make use of wordify since that will make it easier than lab6.

Test these methods on Wikipedia.txt and confirm that you get the same counts as in lab6. Also, turn the Wikipedia.txt file into a single line file called line.txt using lineify, then turn it into a file words.txt with one word per line. Let line2.txt be the file produced by applying lineify to words.txt. Confirm that line.txt and line2.txt are exactly equal as files. All this testing code should occur in the main method.