275 Java Pre-test Practice 2018 Name _____ Login _____

To get the most out of this document:

1. fill it in by hand. Then
2. type the code into your computer and make changes until it runs. Then
3. have a neighbor check your code style, or bring it to office hours. Then
4. write it by hand again taking into account corrections.
5. Repeat until flawless.

1. (0 pts) Any problem from lecture exercises is fair game on the exam.
2. (30 pts) Fill in the class below, making an ArrayList of Cows where appropriate. A Cow has a name. Read the entire program, and read the comments carefully to understand what you have to write. You may not need every line.

```
import java.util.ArrayList;
import java.util.Collection;

class Cow{

    String name; //Use appropriate cow names

    //Write a simple constructor

        _____

        _____

        _____


    //Write toString method

        _____

        _____

        _____


    public static void main(String[] args){

        Collection<Cow>  _____

        //Add two cows to the collection

        _____
```

```
        _____

        //Using a for-each loop, print the entire collection

        _____

        _____

        //Now show a simpler way to print the contents of the collection.

        _____
    }
}
```

3. (20 pts) Complete the main() by adding code below comments. Note that if the constants are updated, your code should still work.

```
   public static void main(String[] args) {

      final int LENGTH = 670; //size of data set to go into array
      final int NUMS_PER_ROW = 30; //numbers per row when printed

      //declare an array to hold the integer data set

      _____

      //Initialize array to the multiples of four {0,4,8,...}

      _____

      _____

      _____

//Print the array with NUMS_PER_ROW elements per row; last row may have fewer.

      _____

      _____

      _____

      _____

      _____

      _____

      _____
}
```

4. (20 pts) Write a method that takes an int array of any size as the only parameter, and returns the sum of its contents.

5. (20 pts) Write a method that takes an ArrayList¡Dog¿ as the only parameter, and returns the total number of legs of all dogs.

6. (20 pts) Write a method that takes an int array of any size as the only parameter, and returns the average of its contents. [1]

7. (20 pts) Write a method that efficiently computes and prints the factorials of 0-9. Hint: you only need seven multiplication ops.

8. (20 pts) Write a method that takes an int array and a single int pivot, and efficiently places all the numbers lower than the pivot at the start of the array, and the ones larger at the end (it partitions the array). Do not sort (because that would be inefficient - why?). Hint: start with an index at each end of the array, and move each index until it is on a number that is in the wrong location. When they are each on such a number, swap the numbers in the array.

9. (20 pts) Write a method that takes an int array of any size as the only parameter, and returns the minimum value of its contents. Your code should be O(n).

10. (20 pts) Write a method that takes an ArrayList¡int¿ as the only parameter, and returns the minimum value of its contents. Your code should be O(n).

11. (10 pts) Consider the following code. What will it print? Briefly explain the difference between reference equality and using the equals() method.

```
class MyStringEquality{

    public static void main(String[] argyle){

        String s1 = "dog";
        String s2 = "cat";
        String s3 = "dogcat";

        String temp  = s1 + s2;

        System.out.println(temp == s3);
        System.out.println(temp.equals(s3));
        //possibly helpful video https://www.youtube.com/watch?v=qQe69w1YF54
    }
}

prints:

explanation:
```

[1] remember that the average of 1 and 2 is 1.5!

12. (20 pts) Write a main() that will perform setup and call one of the methods above (my choice of course).

13. (20 pts) Write a JUnit test for a given method (similar to what you did in lab 1).

14. (20 pts) Write simple Cow and Horse classes. Write a simple interface, and make Cow and Horse implement your interface. Write a main that puts a Cow and a Horse into a Collection, and then calls your method from every object in the Collection.

15. (20 pts) Write simple Car, Cow and Horse classes. Represent that all three move, that cars have wheels, and that cows and horses have hooves. (Hint: you'll need an interface and a super class to do this in a nice, OO way.)

16. (30 pts) Write a Cat class. A cat has a number of legs, a weight, and a name. Write three constructors with parameters (respectively) legs/weight/name, name/weight, and name/legs. Create three cats, place them in a list, and show them printing nicely. Use your best coding practices!!

    In the comments, briefly state a significant problem with this design (from the perspective of a coder using your class).

17. (20 pts) Write code to map the return values from nextInt() (note lack of parameter!) in the Random package into three approximately equal parts. Demonstrate by calling 1,000,000 times, then printing the total number of times a number mapped into each part. Note - you are only printing three numbers.

18. (20 pts) Make a "constant" equal to 12. Declare an array of 12 ints. Initialize the array using a loop, then use a different loop to print the array, showing four lines of three numbers. Now make one change to a single number (your constant) so that the program prints 75 numbers in 25 lines of three numbers (or 90 numbers in 30 lines of three numbers, etc). You should not have to change anything else in the code. [2]

[2] If you did not use the modulus operator for the last two problems, you probably did too much work, and you may have excessive code.