CSCI 111
Final
Fall 2014

<table>
<thead>
<tr>
<th>Question Number</th>
<th>Point Value</th>
<th>Points Awarded</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>20</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>15</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>30</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>15</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>20</td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td>100</td>
<td></td>
</tr>
</tbody>
</table>

Name ___________________________
Student Id ________________________

- 110 Minutes
- Write legibly and keep answers concise.
- You do not need to put comments in any code you write.
- You may use at most one piece of paper (front and back) of notes.
- You may use a laptop to test solutions as well as reference previous labs.
- NO PARTNERS, TELEPHONES, INTERNET (disable your wireless connection).
**Question One - (20 points)** Implement a Dog class for an animal rescue shelter as described: A Dog has a name (such as “Watson”), and weight (such as 39.25). Set the name and weight to values passed in the constructor. Create a “get method” for the instance variables that require it (based on the Driver code below). Create a method that will add the amount of food the dog ate (the decimal weight of food) in that feeding session. Finally, create a method that will calculate and return the average weight of food consumed per feeding (such as .45) by that dog. You do not need to provide any comments in your code.

Hint: You may need instance variables I did not tell you about.

Here is sample code and output from a Driver for your Dog class:

```java
Dog d1 = new Dog("Watson", 39.25);
d1.feed(.15); // d1 was fed .15 pounds of food
d1.feed(.3);  // d1 was fed .3 pounds of food
d1.feed(.7);  // d1 was fed .7 pounds of food
d1.feed(.6);  // d1 was fed .6 pounds of food
System.out.println(d1.getName() + " ate an average of " + d1.getAvgAte() + " pounds per meal");
```

Output (note, 1.75/4 = .4375):

```
Watson ate an average of 0.4375 pounds per meal.
```

Write your class here:
**Question Two (15 points)** – Build a method called `printFactorization` that will take an integer parameter (greater than or equal to 2) and print out the prime factorization of the passed parameter. Nothing is returned, just values printed. You may assume that the method `isPrime(int num)` already exists and will return the boolean value `true` if the passed parameter is prime and `false` if it is not.

The prime factorization of an integer is the list of prime numbers that divide it evenly. For example, the prime factorization of 60 is $2 \times 2 \times 3 \times 5$.

The format and content of the output when `printFactorization(12)` is called:

$$12 = 1 \times 2 \times 2 \times 3$$

The format and content of the output when `printFactorization(5)` is called:

$$5 = 1 \times 5$$
Question Three (30 points) – Build a class that has the following:

- An instance variable to represent a 2d integer array.
- A constructor that will take the number of rows and columns as parameters and create the array.
- A method called `fill` that will fill each spot in the array with a random int from 3 to 121 (including both).
- A method called `search` that will take a parameter and return `true` if it is in the array and `false` if not.
- A method called `searchRange` that will take a lower bound and an upper bound and for each number in that range (including both bounds), print out whether or not that number is in the array.

No instance variables or methods are allowed beyond the ones I indicated above.

```java
public class ArrayStuff {

```
Question Four (15 points) – Build a method called `alphabetize` that will accept a String array and will return that array, now in reverse alphabetical order. You may NOT use any built in Java sorting functions. You may use a method in the String class that has the following header:

```
public int compareTo(String anotherString)
```

The `compareTo` method will return a negative number if the String object that it is called on comes before (in alphabetical order) the String parameter being passed. For example, `a.compareTo("car")` will return `-2` if `a` is the String “apple”. It will return a positive number if the object comes after the parameter and will return `0` if the object and the parameter are the same.

Your method must compile with the following code from a main method:

```java
String[] w = {"pizza", "app", "zebra", "apple"};
String[] orderedWords = Test.alphabetize(w);
for (int i = 0; i < orderedWords.length; i++)
{
    System.out.println(orderedWords[i]);
}
```

The output is:

```
zebra
pizza
apple
app
```
Question Five (20 points) – Answer the following short answer questions.

a. (6 points) Define or give an example of the following (don’t write a dissertation, short answer):
   i. Object Instantiation.
   ii. Object Initialization.
   iii. Variable Declaration.
   iv. Variable Assignment.

b. (1 point) How can you tell a method is recursive?

c. (1 point) What does the access modifier protected mean?

d. (2 points) If $n \% 5 = 0$, what can you say about $n$?

e. (2 points) What are the possible values of $x$ in the statement: `int x = y % 5`?

f. (1 point) What could you use a break statement for?

g. (1 point) What does the keyword super do in Java?

h. (2 points) What does the keyword this mean in Java?

i. (2 points) When is a good time to use a while loop? What about a for loop?

j. (2 points) What is stored in the following variable (be precise)?
   ```java
   GroceryItem milk = new GroceryItem("Milk", 3.09);
   ```