CMSC 132 Quiz 2 Worksheet

The next quiz for the course will be on Wed, Feb 24. The following list provides additional information about the quiz:

- The quiz will be a written quiz (no computer).
- The quiz will be in lab session.
- Closed book, closed notes quiz.
- Answers must be neat and legible.
- Quiz instructions can be found at http://www.cs.umd.edu/~nelson/classes/utilities/examRules.html
- Make sure you know your section number and your TA's name.
- You must take your quiz in your assigned lab/discussion section and not show up to a random discussion section. We will not grade quizzes taken in the incorrect section.
- **Regarding Piazza -** Feel free to post questions in Piazza regarding the worksheet and possible solutions to problems, but for coding questions please do not post code. You can post suggestions on how to solve coding problems, but your classmates will benefit more if they themselves actually solve the problems. Pretend you are a TA while addressing or providing help in Piazza ☺

The following exercises cover the material to be included in this quiz. Solutions to these exercises will not be provided, but you are welcome to discuss your solutions with the TAs or instructors during office hours. It is recommended that you try these exercises on paper first (without using a computer).

Exercises

1. You should be familiar with Object-Oriented Design similar to the lab exercise:

   http://www.cs.umd.edu/class/spring2016/cmsc132-04XX/content/labs/Week4/DesignExercise.pdf

2. Give the asymptotic bound of the following functions:

   a. $\mathbf{n}^3 + \log(\mathbf{n})$ — f(**n**) = O( )
   b. $\mathbf{n}^2 - 200\mathbf{n} - \mathbf{n}^2$ — f(**n**) = O( )
   c. $2\mathbf{n}^4 - 500\mathbf{n} - \mathbf{n}^2$ — f(**n**) = O( )

3. List the following big-O expressions in order of asymptotic complexity (lowest complexity first).

   O(nlog(n)) O(log(n)) $O(n^2)$ $O(n^3)$

4. Implement the iterator() method for the class **Evens** below. The method will return an iterator that will allow us to retrieve even values starting at 2 and ending at the maximum value (including it) defined by the instance variable **max.** The main and expected output provided below illustrates the functionality associated with the iterator. For this problem:

   a. You must use an **inner class** to implement the iterator.
   b. You do not need to implement the remove method.

```
public class Evens implements Iterable<Integer> {
       private int max;

       public Evens(int max) { this.max = max; }

       public static void main(String[] args) {
              Evens evens = new Evens(10);
              for (Integer i : evens) {
                     System.out.print(i + " ");
              }
              System.out.println();

              Evens evensTwo = new Evens(20);
              for (Integer i : evensTwo) {
                     System.out.print(i + " ");
              }
       }
}
```

```
Output

2 4 6 8 10
2 4 6 8 10 12 14 16 18 20
```

5. A Bag is similar to a set except it we can keep track of how many instances have been added for a particular element. For this problem:

    a. Implement a Bag class that keeps track of strings added to the bag.
    b. The following methods should be supported:

        i. void add(String elem) → adds element to the bag.
        ii. int getCount(String elem) → returns the number of instances associated with elem.
        iii. String toString() → prints each string and number of instances for that string.
        iv. Define an iterator for the bag that allow us to go through all instances of the bag. For example, if the bag has two instances of the string "cat", it will take two calls to next() in order for us to retrieve everything the bag has. You must use inner classes for the iterator.
        v. Define the remove method of the iterator interface. The method will reduce the number of instances of a particular string by one.
        vi. Feel free to add any other methods/instance variables you understand are necessary.