This exam has 11 questions, for a total of 200 points.

<table>
<thead>
<tr>
<th>Question</th>
<th>Points</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>10</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>6</td>
<td>20</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>20</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>15</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>30</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>25</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>25</td>
<td></td>
</tr>
<tr>
<td>Total:</td>
<td>200</td>
<td></td>
</tr>
</tbody>
</table>
1. (10 points) **Polymorphism.** Given these classes and interfaces, which of the following assignments are valid? If invalid, explain why.

```java
class BlueThing { ... }
interface Singer { ... }
class Smurf extends BlueThing implements Singer { ... }
class Turf extends BlueThing { ... }
```

a. BlueThing b = new Smurf();

b. Singer s = new Singer();

c. Turf t = new BlueThing();

d. Singer s = new Smurf();

e. Turf t = new Smurf();

2. (10 points) **Abstract Data Type.** What is meant by the term “Abstract Data Type”? How are Abstract Data Types specified in Java? Give an example.
3. (10 points) **Tracing Queue Operations.** Trace a queue myQ through the following operations. Show the state of the queue after each call.

```java
myQ.enqueue(new Integer(3));
myQ.enqueue(new Integer(1));
Integer x = myQ.dequeue();
myQ.enqueue(new Integer(1));
myQ.enqueue(new Integer(3));
myQ.enqueue(new Integer(8));
Integer y = myQ.first();
Integer z = myQ.dequeue();
myQ.enqueue(new Integer(2));
myQ.enqueue(new Integer(9));
```

4. (15 points) **Loop Analysis.** Given a populated LinkedList<String> called allCitizens, compare the run times of the following three loops:

a. for (int i = 0; i < allCitizens.size(); i++) {
   System.out.println( allCitizens.get(i) );
}

b. Iterator it = allCitizens.iterator();
   while ( it.hasNext() ) {
      System.out.println( it.next() );
   }

c. for ( String citizen : allCitizens ) {
   System.out.println( citizen );
}
5. (20 points) **Short Answer.** Respond to each of the following True or False statements. Explain your answers.

   a. An ArrayList will work anywhere a LinkedList would work.

   b. The user of a LinkedList needs to understand how to work with Nodes.

   c. The user of an ArrayList needs to understand arrays.

   d. An OrderedList is an Abstract Data Type (ADT).

   e. A $O(n^2)$ sorting algorithm should never be used when there are $O(n \log n)$ sorting algorithms to choose from.
6. (20 points) **Short Answer.**

   a. Under what conditions is adding or removing an element in a linked structure a O(1) operation?

   b. Under what conditions is adding or removing an element in an array-based structure a O(1) operation?

   c. How does the memory usage of an array-based structure compare to a linked structure?

   d. What structure / implementation would be the best choice for managing a collection where order is not important, but the user needs frequent access to elements within the collection? Explain your choice.
7. (20 points) **Method Analysis.** Determine the Big-O runtime of the following method.

Show the contents of numbers after calling bar() if it initially contained \(\{2,3,4\}\).

Show the contents of numbers after calling bar() if it initially contained \(\{1,2,3\}\).

Is the Big-O runtime affected by the length of numbers, the values it contains, or both?

```java
public void bar(int[] numbers) {
    for (int idx1 = numbers.length - 1; idx1 >= 0; idx1--) {
        for (int idx2 = 0; idx2 < idx1; idx2++) {
            if (numbers[idx1] % 2 == 0) {
                int tmp = numbers[idx2];
                numbers[idx2] = numbers[idx2 + 1];
                numbers[idx2 + 1] = tmp;
            }
        }
    }
}
```
8. (15 points) **Getting Data from a List.** Given the following method

```java
Integer foo(List<Integer> a) {
    Integer sum = new Integer(0);
    for (int x = 0; x < a.size(); x++) {
        sum += a.get(x) * a.get(a.size() - x - 1);
    }
    return sum;
}
```

a. What prints out for a call to `System.out.println(foo(list))` if the list contains `{1,2,3,4}`?

b. What is the Big-O if we call `foo(list)` where the list is a LinkedList with N elements in it? Explain.

c. What is the Big-O if we call `foo(list)` where the list is an ArrayList with N elements in it? Explain.
9. (30 points) Given the following framework for a singly-linked list, write the add(T element) and remove(T element) methods. You should assume the Node class methods are fully functional. remove() may throw an unchecked ElementNotFoundException. (The problem continues on the next page.)

```java
public class BestListEver<T> {
    private Node head = null;
    private int size = 0;

    private class Node {
        private Node next = null;
        private T element = null;

        public Node getNextNode { ... }
        public void setNextNode { ... }
        public T getElement { ... }
        public void setElement { ... }
    }

    public void add(T element) {
    }
}
```
10. (25 points) Write a Java method that reverses the order of elements in an array of objects using a Stack. You may assume the API Stack class is implemented with the standard Stack methods.

What is the runtime of your method?
11. (25 points) Given the following class definition for a node in a doubly-linked list:

```java
Class Node {
    Object data;
    Node previous;
    Node next;
}
```

Write a recursive method with the signature

```java
int calcSize(Node n);
```

that calculates the number of nodes in the list beginning at Node n.

What is the Big-O runtime of this method?
scratch paper
scratch paper