CSCI 2320
Fall 2018
Exam 3

Last Name: _____

First Name: _____

Write your answers on the answer sheets provided unless specified otherwise. You may refer to printouts of the Java Command Sheets provided at the beginning of the semester.

```
class BaseballPlayer {
   private String name;
   private int atBats;
   private int baseHits;
   private int rbis;
}
class ListNode {
   int data
   ListNode next;
}
class MyList {
   private ListNode head;
   public MyList() { head= null; }
   public void add(int elem) { ... /* adds to front of list */ }
   public void remove(int elem) { ... /* removes specified element list */ }
}
class MyStack {
   // internal implementation not specified, but does have these methods
   public void push(char elem) {}
   public char pop() {}
   public boolean isEmpty() {}
}
```

1. (4 pts) Expand the acronym FIFO. What data structure is it associated with?
2. (2 pts each) Answer these questions about overriding `.equals()` for a custom class in Java.
   (a) Why is overriding `.equals()` sometimes necessary?
   (b) What are some issues a programming might encounter when overriding `.equals()`?
3. Answer these questions about the `BaseballPlayer` class defined in the code above.
   (a) (6 pts) Write a well-formed `.compareTo()` method that will indicate one player is greater than another if they have more RBIs than another player.
   (b) (4 pts) Show how the quicksort partition method below would be changed if it were to be used to sort `BaseballPlayer` objects rather than integers. Write your answer on this exam sheet by writing in the changes rather than by rewriting the the entire method.

```
public static int partition(int [] wordList, int start, int stop)
{
   int left, right;
   int temp;
   int k= wordList[start];

   left= start-1;
   right= stop+1;

   do {
      do { right--; } while (wordList[right] > k);
      do { left++; } while (wordList[left] < k);

      if (left < right)
      {
         temp= wordList[left];
         wordList[left]= wordList[right];
         wordList[right]= temp;
      }
   } while (left < right);
   return right;
}
```

4. (4 pts) In the previous problem the original Quicksort partition code picked the first element in the partition to be the "partition element". Suppose we modified it so that instead we picked the middle element in the partition to be the partition element (i.e., `k= wordList[(start+stop)/2];`). Would the sort still work? Explain.

5. (6 pts) Below is an iterative implementation of a binary search routine that searches for `searchElem` in an array (`a`) of doubles starting at position `start` and ending at position `stop`. If the element is not found the function returns -1. Modify the function to be recursive. The function should achieve the exact same result as the original, however.

```
public static int binSearch(double [] a, int start, int stop, double searchElem)
{
  int mid= (start+stop)/2;
  while (start<=stop) {
    if (a[mid]==searchElem) return mid;
    if (a[mid]<searchElem)
      start= mid+1;
    else
      stop= mid-1;
    mid= (start+stop)/2;
  }
  return -1;
}
```

6. (4 pts) Joe says: "Binary search is waaaaay better than a sequential search. Sequential search is for baby programmers who haven't learned how to do a binary search. Anyone who uses a sequential search clearly doesn't know squat about programming!" (Clearly Joe is compensating for something). Do you agree with Joe or not. Explain your answer.

7. Answer these questions about linked lists and the `MyList` class given above.

   (a) (6 pts) Write a `size()` method for the `MyList` class that will return the number of elements in the `MyList` object. We do *not* keep the list size in a variable so you will need a loop.

   (b) (6 pts) Write a method for the `MyList` class called `isSorted()` that will return true if the elements in the list are ordered from least to greatest; false otherwise. An empty list is considered to be sorted.

   (c) (10 pts) In statistics the "mode" is the element that appears most often in a data set. Suppose that the elements in `MyList` are ordered from least to greatest. Write a method called `findMode()` that will return the mode of the list. If there is more than one mode then return the first element in the list meeting that requirement. If the list is empty return -1 for the mode.

8. Answer these questions about stacks and the `MyStack` class given above.

   (a) (4 pts) Suppose that when a letter is encountered we push it on the stack and when an asterisk is encountered we pop the stack. Show the sequence in which letters would be popped for the following string:

   ```
   EW*DALE**CODL***EN***RO*M*****
   ```

   (b) (10 pts) Write a method called `isValid()` that accepts a parameter string containing various characters including, among other things, parentheses and square brackets. Assuming that the `MyStack` class has been properly implemented, use it to identify whether the grouping characters are properly matched in the string. To do this, when you encounter a left grouping character, push it on the stack. When you encounter a right grouping character, pop the stack and make sure the popped character is of the same type as the one you are matching. If the characters don't match or if the stack is empty when you encounter a closing character or if the stack is not empty when you are done parsing the string then the string is not valid; otherwise it is valid. Some examples:

`[abc(def)]`	valid
`[abc(def])`	not valid
`((abc[def]))`	valid
`((abc[def])`	not valid
`)(`	not valid

9. Answer these questions about queues.

   (a) (4 pts) Name advantages and disadvantages of using linked lists or arrays as a basis for implementing a queue.

   (b) (4 pts) Suppose that when a letter is encountered we put it in the queue and when an asterisk is encountered we remove from the queue. Show the sequence in which letters would be removed for the following string:

   ```
   EW*DALE**CODL***EN***RO*M*****
   ```

10. (4 pts) Which homework assignment in this course was your favorite? Why?

11. (4 pts) Which topic in this course was your favorite?