1. (15) Fix the following Java code snippets. Note that there may or may not be multiple errors in each snippet. There will always be at least one thing wrong with each snippet. I'm not that sneaky.

   a. The following is supposed to find the minimum value in an integer array:

```
//myArray is an integer array declared and populated above
int currentMinimum = myArray[0];
for (int i=1; i<myArray.length; i++) {
     if (myArray[i] < currentMinimum) {
          currentMinimum = myArray[i];
     }
}
```

   b. The following method is supposed to return the index of a given value in an integer array:

```
int findIndex(int[] myArray, int value) {
   int indexVal = -1;//return -1 if value isn't found
   for (int i=0; i<myArray.length; i++) {
          if (myArray[i] == value) {
               indexVal = i;
          }
   }
   return indexVal;
}
```

   c. The following method is supposed to compute the area of a circle given a radius using two different values for π:

```
void computeArea(double radius) {
   double pi = 3;
   double area = pi * radius * radius;
   System.out.println("A poor area estimation is " + area);
   pi = 3.14159;
   area = pi * radius * radius;
   System.out.println("A better area estimation is " +area);
}
```

2. (15) State the runtime for the following pieces of code in terms of the input size, n. You can phrase this as a sentence or using Big-Oh notation.

a. The following code to sort an array of doubles:

```
//myArray is a double array declared and populated above
double temp;
for (int i=0; i<myArray.length; i++) {
     for (int j=i+1; j<myArray.length; j++) {
          if (myArray[i] < myArray[j]) {
               temp = myArray[i];
               myArray[i] = myArray[j];
               myArray[j] = temp;
          }
     }
}
```

O(n^2)

b. The following code to search for duplicates in an array of doubles:

```
bool isUnique(double[] myArray) {
     for (int i=0; i<myArray.length; i++) {
          for (int j=0; j<myArray.length; j++) {
               if (myArray[i] == myArray[j] && i != j) {
                    return false;
               }
          }
     }
     return true;
}
```

O(n^2)

c. The following code to search for duplicates in an array of doubles:

```
bool isUnique(double[] myArray) {
     for (int i=0; i<myArray.length; i++) {
          for (int j=i+1; j<myArray.length; j++) {
               if (myArray[i] == myArray[j]) {
                    return false;
               }
          }
     }
     return true;
}
```

O(n^2)

3. (15) Consider the following stack (Abed is at the top, Carmen at the bottom) called myStack:

Abed
Sally
Maria
Thomas
Julius
Carmen

a. What would the stack look like after the following operations (include your work in case of wrong answer for partial credit):

```
String x = myStack.pop();
myStack.top();
x = myStack.pop();
myStack.push(myStack.pop());
myStack.push("Big Joe");
```

Big Joe
Maria
Thomas
Julius
Carmen

b. What is the *best-case* runtime for part (a) as a function of stack size, n?

O(1)

4. (20) Would you implement the following with a singly linked list, a doubly linked list, or an array and why?

    a. Implementing a stack of user-input text strings.

    Singly linked list because you don't know the size ahead of time and you only need access to one end.

    b. Keeping records of test scores for a class sorted from lowest to highest. Frequent access needed to sort, search, and find the minimum, maximum, and average values.

    Array. You know the size ahead of time, and you want to be able to search a sorted data structure.

    c. Keeping track of a user's most recently entered commands, up to a maximum of 20.

    Array because you know the size ahead of time.

    d. Recording the `Students` currently enrolled in a course from the time enrollment opens until the final drop date.

    Singly linked list because the size varies

5. (15) Write the code for the following methods that work on an array of doubles (you can assume the array isn't empty and is sorted from lowest value to highest):
   a. findMean, to find the average (arithmetic mean) of an array.

```
double findMean (double[] myArray) {
      double mySum = 0;
      for (int i=0;i<myArray.length;i++) {
           mySum = mySum + myArray[i];
      }
      return mySum/myArray.length;









}
```

   b. findMedian, to find the median (value in the "middle" of a sorted) of an array. Should return the mean of two most central values if no true median exists (if there are an even number of entries in the array).

```
double findMedian (double[] myArray) {
      if (myArray.length % 2 == 1) %If the length is odd
           return myArray[(myArray.length-1)/2];
      else {
           double returnVal;
           returnVal = (myArray[myArray.length/2] + myArray[myArray.length/2 - 1])/2;
           return returnVal;
      }

}
```

   c. findMode, to find the value the occurs the most in the array. This will likely be the most challenging of the three.

```
double findMode (double[] myArray) {
      int maxCount = 0;
      int currentMode;
      int currentCount;
      for (int i=0;i<myArray.length;i++) {
           currentCount = 1;
           for (int j=i+1;j<myArray.length;j++) {
                if (myArray[j] == myArray[i])
                     currentCount = currentCount + 1;
           }
           if (currentCount > maxCount) {
                maxCount = currentCount;
                currentMode = myArray[i];
           }
      }
      return currentMode;
}
```

6. (15) Convert the following code to work using Java generics to store the element instead of requiring integers. Explain why this is advantageous.

```
public class Node <E> {

      public E element;
      public Node next;

      public Node(E e, Node n) {
            element = e;
            next = n;
      }

}
```

This is advantageous because it allows us to write a single node class that can then be used to store nodes of ints, chars, Strings, doubles, Students, etc without having to write a new node class each time. This is particularly useful if you aren't going to know ahead of time what type(s) the user of your data structure will be working with.