CSC 20: Practice Exam (Spring 2018)

Student Name:_____

Directions - please read the questions carefully. Pace yourself by keeping track of how many problems you have left to go and how much time remains. You do not have to answer the problems in any particular order, so move to another problem if you find you are stuck or spending too much time on a single problem. Please write precise and clean answers.

1. True/False questions

Statement	True	False
1) The expression "Java " + 1 + 2 + 3 evaluates to Java 6		X
2) Class B inherits from Class A, B has access to members of A.	X	
3) An Abstract Data Type (ADT) must obviously have some kind of representation for its data. We often make this presentation public so that a user can change it to improve our coding.		X
4) In Java, we pass parameters to methods as references.	X	
5) Class and object are just different names for the same thing.		X
6) The following code is an example of instantiating a String object: String str = String("Hello");	X	

7) The following code:
if (condition) { do A; do B; do BC; }
else { do A; do D; do DC; }
can be **optimized** with:
do BC; do DC;
if (condition) { do A; do B; }
else { do A; do D; }
		X

8) The different versions of an overloaded method are differentiated by their signatures (types).	X	
9) An advantage of arrays compared to linked lists is that it takes fewer steps to insert a new element at the beginning of a long array than at the beginning of a long linked list.	X	
10) Consider the statement "x = (a > b) ? a : b"; then the value of x is 27, if a = 18 and b = 27.	X	

2. Arrays, Functions.

The following two methods do the same job. They each take an ORDERED array of ints and a target number as arguments.

```
public static boolean mystery1(int[] array, int target) {
   for (int i = 0; i < array.length; i++) {
           if (array[i] == target)
              return true;
           else if (array[i] > target)
              return false;

    }
    return false;
}

public static boolean mystery2(int[] array, int target) {
    int low = 0;
    int high = array.length - 1;
    while (low <= high) {
       int mid = (low + high) / 2;
       if (array[mid] == target)
         return true;
       else if (array[mid] < target)
         low = mid + 1;
       else
         high = mid - 1;
    }
    return false;
  }
```

Use this array to answer the following questions:

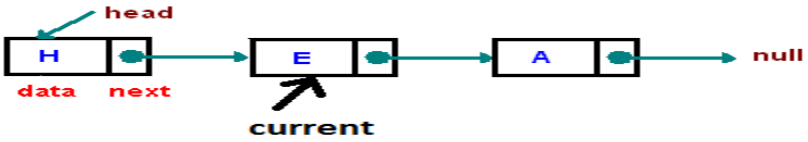
int[] a = { 2, 5, 11, 14, 15, 27, 31};

a) What does mystery1(a, 5) return? (5 points – No partial credits)

b) Fill in the trace table to show that mystery2(a, 5) returns the same thing. (10 points)

target	low	hi	mid	return value

3. Data Structure: Consider a LinkedList object's lst below (a starter point):



Draw diagrams to show the LinkedList lst **after EACH** of the following operations are performed: (1) lst.addFirst("B"), (2) lst.addLast("C"), (3) lst.insertAfter("M"), (4) lst.backward(), (5) lst.removeFirst() . Please be sure to label the head, node's data, current, and nodes's next pointers (10 points). Note: Please draw your diagrams using the data from the previous step as you go through each operation: starting from a starter point, to (1), (2), (3), (4), and to (5).

4. **Object Oriented Programming: Implement a new method for the LinkedList class (lab 6). This method performs a swapping of the head and the tail nodes in a list where it has more than 1 element. The current position is required to swap if it is pointing to the head or tail node. The method does not require input argument(s). If the operation is performed successfully, the method will return a Boolean true value. Otherwise, it returns a false value. Provide the Java code for this method.**

Hint: For a starter, you might consider explaining your idea in a diagram accompanied with pseudo code (in English). Then finally translate this work into a Java method. Please use the back of this page as needed.

```
public boolean swapHeadAndTail() {

}
```

5. More on Arrays, Functions.

Implement a method that reverses elements in the array provided as an argument.

```
public class ReverseEelements {
    public static void reverse(double[] a)
   {
     // write your code here









  }
  public static void main(String[] args) {
      double[] a = {1.1, 5.3, 3.4, 8.9, 0.0, 1.3, 5.2, 7.8, 9.9};
      System.out.println("initial array = " + Arrays.toString(a));
      // call reverse method
      reverse(a);
      System.out.println("reversed array = " + Arrays.toString(a));
   }
}
```

Example program output:

```
initial array = [1.1, 5.3, 3.4, 8.9, 0.0, 1.3, 5.2, 7.8, 9.9]
reversed array = [9.9, 7.8, 5.2, 1.3, 0.0, 8.9, 3.4, 5.3, 1.1]
```