Intro. to Computer Science: Test 1

Short Answer Questions

1. (a) What is the output of the following program?

```java
public class ArgumentTest{
    public static void main(String [] args){

        int i=5, j=9;
        int [] a = {1,2,3,4};

        System.out.println("line 1: i = " + i + " j = " + j);  

        messUp(a,i,j);

        System.out.println("line 5: i = " + i + " j = " + j);  
    } /* end main */

    public static void messUp(int [] a,int i,int j){

        a[2] = i;
        j = a[3];
        i = a[1];
        System.out.println("line 3: i = " + i + " j = " + j);
    } /* end messUp */

} /* end class ArgumentTest */
```

line 1: i = 5 j = 9
line 2: a[2] = 3
line 3: i = 2 j = 4
line 5: i = 5 j = 9
(b) (i) What is the output of the following (incorrectly indented) code fragment when \(x = 2\) and \(y = 3\).

(ii) What is the output if \(x = 3\) and \(y = 2\)?

\[
\begin{align*}
&\text{if (x > 2)} \\
&\quad \text{if (y > 2)} \{ \\
&\quad\quad \text{int z = x + y; } \\
&\quad\quad \text{System.out.println(" z is " + z); } \\
&\quad\} \\
&\quad \text{else} \\
&\quad \text{System.out.println("x is " + x);}
\end{align*}
\]

When \(x = 2\) the first ‘if’ clause is false, and nothing prints. When \(x = 3\), the ‘if’ clause is true, so the test if \((y > 2)\) executes. Since this is false, the else clause executes, printing “x is 3”.

(c) What is the output of the following loop:

\[
\begin{align*}
&\text{for (int i = 1; i < 4; i++)} \{ \\
&\quad \text{for (int j = 1; j < 4; j++)} \{ \\
&\quad\quad \text{if (i*j > 2) continue;} \\
&\quad\quad \text{System.out.println(i*j);} \\
&\quad\} \\
&\quad \text{System.out.println(i);} \\
&\}
\end{align*}
\]

The \(i,j\) pairs in the for loops are \((1,1),(1,2),(1,3),(2,1),(2,2),(2,3),(3,1),(3,2),(3,3)\). Of these, the test if \((i*j > 2)\) will skip the println statement for \((1,3),(2,2),(2,3)\) and all the \(i=3\) pairs. The continue statement only skips to the next iteration of the inside (nested) loop; each iteration will still print the \(i\) before the next outer iterations.

So the output will be: \(1\ 2\ 1\ 2\ 2\ 3\) (with a new line between each number).
2. Write a method called `findMin` that takes an array of doubles as an argument, and returns the index of the smallest number in the array. If there are duplicates, then it should return the index of the last occurrence.

```java
public static int findMin(double[] a)
{
    int minIndex = 0; // initialize to element of array

    for (int i = 0; i < a.length; i++)
    {
        if (a[i] <= a[minIndex]) // include 'equals' to get last occurrence
            minIndex = i;
    } // end for i

    return minIndex;
} // end findMin
```

3. Write a program that takes as input a string representing a hexadecimal number, and converts it to decimal. You should use the `charAt` method for strings to get each individual hex character out of the string.

```java
import java.util.Scanner;

public class Convert2Decimal {
    public static void main(String args[])
    {
        Scanner input = new Scanner(System.in);

        System.out.print("Enter hex string: ");
        String s = input.next();

        int decVal = getDecimal(s); // function call with one argument
        System.out.println("Decimal equivalent is " + decVal);
    } /* end main */

    public static int getDecimal(String s)
    {
        int len = s.length();
        int dec = 0, digit = 0;

        for (int i = 0; i < s.length(); i++)
        {
            char c = s.charAt(i);

            if (c >= 'A' && c <= 'F') digit = c - 'A' + 10;
```
else if (c >= 'a' && c <= 'f') digit = c-'a'+10;
else if (c>='0' && c <= '9') digit = c-'0';
else {
    System.out.println("found non hex digit ");
    System.exit(0);
}

dec += digit*Math.pow(16,len-i-1);
} /* end for i */
return dec;
} /* end getDecimal */

} /* end Class Convert2Decimal */

The next two questions both involve writing a method reverseArray to reverse the elements of an array of characters. They will use two different approaches.

4. Write a method reverseArray that takes as input an array of characters, and returns a reference to a new array with the elements reversed. In other words, in this first approach you should use a second array in your method.

public static char[] reverseArray(char [] a){
    char [] b = new char(a.length);
    int i,j;

    j = a.length-1;
    for (i=0;i<a.length;i++){
        b[j] = a[i];
        j--;
    } // end for i

} /* end reverseArray */

5. In this version of reverseArray you do not make a copy of the array, but do it in place. (In other words, do not make another array, you should only use one or two additional variables). The method does not return any arguments.
```
public static void reverseArray(char [] a){
    int j;
    int halfLength = a.length/2; // works for even and odd lengths
    j = a.length-1;
    for(int i=0;i<halfLength;i++){
        char tmp = a[i]; // need extra variable for swapping
        a[i] = a[j];
        a[j] = tmp;
        j--;
    }/* end for i */
}/* end reverseArray */
```

6. **Extra Credit**: only if you have time. Write a program that reads integers, find the largest of them, and counts its occurrences. Assume the input ends when the number 0 is entered. For example, if \(3 \ 5 \ 2 \ 5 \ 5 \ 5 \ 0\) is entered, the program should report that the largest is 5 and the occurrence count for 5 is 4.

```java
import java.util.Scanner;

public class CountInts{
    public static void main(String[] args){
        int count=0;
        int maxFound, nextNum;
        Scanner input = new Scanner(System.in);
        System.out.print("enter an integer, 0 to quit ");
        maxFound = input.nextInt(); /* initialize max to first int */
        if (0 == maxFound){
            System.out.println("no numbers entered");
            System.exit(0);
        }
        else count=1; /* the number entered has occurred once so far */
        System.out.print("enter an integer, 0 to quit ");
        while ((nextNum= input.nextInt()) != 0){
            if (nextNum == maxFound)
                count++;
            else if (nextNum > maxFound){
                maxFound = nextNum;
                count = 1;
            }
```
System.out.print("enter an integer, 0 to quit ");
} /* end while */

System.out.println(" max number was " + maxFound + " repeated " + count + " times ");

} // end main

} // end class CountInts

} // end class CountInts