[CSE2] Final Exam Fall 2014 Name_____

Lehigh Username (Write in upper case letters): | | | | | | | @lehigh.edu

Put your name on each page, or risk having an incompletely graded exam. Raise your hand to notify a proctor if you have a question. **No talking, no tapping, no other noise. No cell phones, calculators or other electronics. No paper, no books, no notes.**

Read the entire exam and answer the questions that seem easiest first. Write answers in the box provided. Answer length/difficulty is not related to box size.

PROFANITY OF ANY KIND WILL RESULT IN A ZERO SCORE.

Grader Use Only

Question 1:	/30	Question 9:	/30
Question 2:	/20	Question 10:	/20
Question 3:	/20		
Question 4:	/20		
Question 5:	/30		
Question 6:	/30		
Question 7:	/20		
Question 8:	/30		
		Exam Total:	

Special use only: This is an:

Anticipatory exam___ Challenge exam___

[CSE2] Final Exam Fall 2014 Name_____

Lehigh Username (Write in upper case letters): | | | | | | | @lehigh.edu

total score

1) Below there is a sequence of statements that exist in the same scope. In the box indicated by the arrow, **state the value** assigned to the variable in the statement just above each arrow. (3 points each, 30 points total)

```
String out="ab", one=1+2*3+"c";
out+=one+2+3*4;
```
→ out =

```
out= "a" + 2+33/4/2%5;
```
→ out =

```
out+=(int)Math.random()*0;
```
→ out =

```
String sout="\\\"hello\"\\";
```
→ sout =

```
out+=(int)(53/12*1.5%20);
```
→ out =

```
String fout="one",two="on"+'e';
boolean bout = fout == two;
```
→ bout =

```
int [][] x={ {12,2,2},{2,3,4},{12,15}};
sout=" "+x[0][2]+x[2][0];
```
→ sout =

```
sout=""+(5%6==0 || 4<5 && 2*3==6);
```
→ sout =

```
bout = true;
bout = bout && 2*3 == 15-8==false;
```
→ bout =

```
sout="";
String temp="IbqqzIpmmjezt";
for(int j=0;j<temp.length();j++){
   sout+=(char)(temp.charAt(j)-1);
}
```
→ (After the loop has fully executed) sout =

Lehigh Username (Write in upper case letters): | | | | | | | @lehigh.edu

total score

2) I showed the code below to my friend. It compiles without error. She pointed out that the code had the potential for four run-time errors, **when given certain input**. Using the line numbers added for your reference, **state the line number**, **explain the error**, and **state what the input would have to be** to cause each error. (5 points each, 20 points total)

```
 1 | public static void runTimeErrs(){
 2 |   Scanner scan=new Scanner(System.in);
 3 |   int list[]=new int[10],count=0,x,sum=0;
 4 |   System.out.print("Enter positive ints , <0 to quit");
 5 |    do{
 6 |      x=scan.nextInt();
 7 |      if(x>=0){
 8 |       list[count]=x;
 9 |       count++;
10 |     }
11 |    }while(x>=0);
12 |    System.out.println("You entered: ");
13 |    do{
14 |      count--;
15 |      System.out.print(list[count]+" ");
16 |      sum+=1/list[count];
17 |    }while(count>0);
18 |    System.out.println("\nThe mean of the reciprocal: "+sum);
19 | }
```

Line #	What input causes it?	Why is this an error?
Line #	What input causes it?	Why is this an error?
Line #	What input causes it?	Why is this an error?
Line #	What input causes it?	Why is this an error?

Lehigh Username (Write in upper case letters): | | | | | | | @lehigh.edu

total score

3) What is printed when the method quizzical(), below, is called? (20 points)

```
public static void quizzical(){
  int x[][]={ {1,2},{3,4} }, y[]={0,1,0};
  int z[][]={ {1,2}, {3,4}};
  enigma('a',3);
  enigma('b',2);
  enigma('c',3);
  System.out.println();
  print(x);
  puzzle(x,y,true);
  print(x);
  print(z);
  puzzle(z,y,false);
  print(z);
}


public static void puzzle(int [][]p,
      int[] q, boolean choice){
  if(choice){ p[0]=q; }
  else { p[q[0]][q[1]]=42; }
}


public static void print(int[][]x){
  for(int row=0;row<x.length;row++){
    for(int col=0;col<x[row].length;col++){
      System.out.print(x[row][col]+" ");
    }
  }
  System.out.println();
}


public static void enigma(char a,int j){
  while(true){
    switch(a){
      case 'a': System.out.print('a');
      case 'b': System.out.print('b');
      default:
    }
    System.out.print("x!");
    j++;
    if(j>5){
      return;
    }
  }
}
```

Answer:

Lehigh Username (Write in upper case letters): | | | | | | | @lehigh.edu

total score

4) Write a method called **pad()**, which takes as input a single dimensional array of Strings, and modifies the entries in the array so that each String has the same length as the longest string. First, find the length of the longest String. Then append blank characters to all Strings that are shorter. **pad()** generates no output (20 points)

Example:

```
String thai[]={"one", "fine", "evening"};        //input
String thai[]={"one    ", "fine   ", "evening"}; //output
```

Lehigh Username (Write in upper case letters): [ ][ ][ ][ ][ ][ ] @lehigh.edu

total score

5) Recall that a 3D array is organized into slabs, rows, and columns, where the slabs are accessed by the first index, the rows are accessed by the second index, and the columns are accessed by the third index. Write the method **ragged()**, which takes as input a 3D array and returns false if and only if each slab has the same number of rows and each row has the same number of columns. For example, for the array v below, ragged(v); returns false. (30 points)

```
int [][][]v = {
{ {2,2,2}, {3,3,3}, {1,1,1}, {2,4,2} },
{ {1,2,1}, {1,3,1}, {2,4,2}, {1,3,1} }
}
```

Lehigh Username (Write in upper case letters): | | | | | | | @lehigh.edu

total score

6) Write a method called **clockwise()** that accepts as input a two dimensional rectangular array of integers, A[], and generates, as output, a two dimensional array of integers. A[] represents a rectangular matrix in the **row major** format: The member array A[0] is the top row of the rectangular matrix and A[A.length-1] is the bottom row. That means the height of the matrix is A.length. For any row i, A[i][0] is the leftmost member of that row, and A[i][A[i].length-1] is the rightmost member of that row. That means the width of the matrix is A[i].length.

The output of clockwise should be a matrix B, also in row major format, that is identical to A[], except that it has been rotated 90 degrees clockwise. (30 points)

Example:
Let this be A:

0 1 2
3 4 5
6 7 8
9 10 11

Then
A[0][0] = 0,
A[0][1] = 1,
A[3][0] = 9,
etc.

Let B =
clockwise(A);

Then B looks like this:

9 6 3 0
10 7 4 1
11 8 5 2

Where
B[0][0] = 9,
B[0][1] = 6,
B[0][3] = 0,
B[2][0] = 11,
B[2][1] = 8,
etc.

Lehigh Username (Write in upper case letters): | | | | | | | @lehigh.edu

total score

7) Write a method **increasing()** that prompts (and forces) the user to input 10 integers that are each as large as the previous entry. The user should be prompted to enter each integer individually (i.e. use a loop). If, at any point, the user provides inputs that do not satisfy these conditions (i.e. that the input is an integer and that it is larger than the last number), print out an error and re-prompt the user for an acceptable number, until they enter one that is. **increasing()** should return the 10 ints entered in an array. Assume Scanner has been imported. Just write the method, no main method. Hint: The Scanner method hasNextInt(), returns true if the user has entered an int. (20 points)

[CSE2] Final Exam Fall 2014 Name_____

Lehigh Username (Write in upper case letters): | | | | | | | @lehigh.edu

total score

8) Rewrite the code in the boxes below with the type of loop indicated below, while making the same calls to the function **x()** or the same changes to the array **A[]**. Use the boxes designated by each letter. (7.5 points each, 30 points total)

a) Rewrite as a do-while loop
b) Rewrite as a while loop
c) Rewrite as a for-each loop.
d) Rewrite as a for-loop

```
int[] a = {2,3,4,5}

for( int val; a){
        System.out.println(val);
        val = 5;
}
```

a)

```
for(int j=a; j<b; j++){
     x(j);
}
```

b)

```
int j=a;
do{
     x(j); j++;
}while(j<b);
```


The
on the same page as the questions, though they may run onto the next.

Lehigh Username (Write in upper case letters): @lehigh.edu

c)

```
int A[] = {1,2,3,4};
for(int i = 0; i<A.length; i++){
     A[i]++;
}
```

d)

```
j=a;
do{
     x(j);
     j--;
} while(j>b);
```

Lehigh Username (Write in upper case letters): [ ][ ][ ][ ][ ][ ] @lehigh.edu

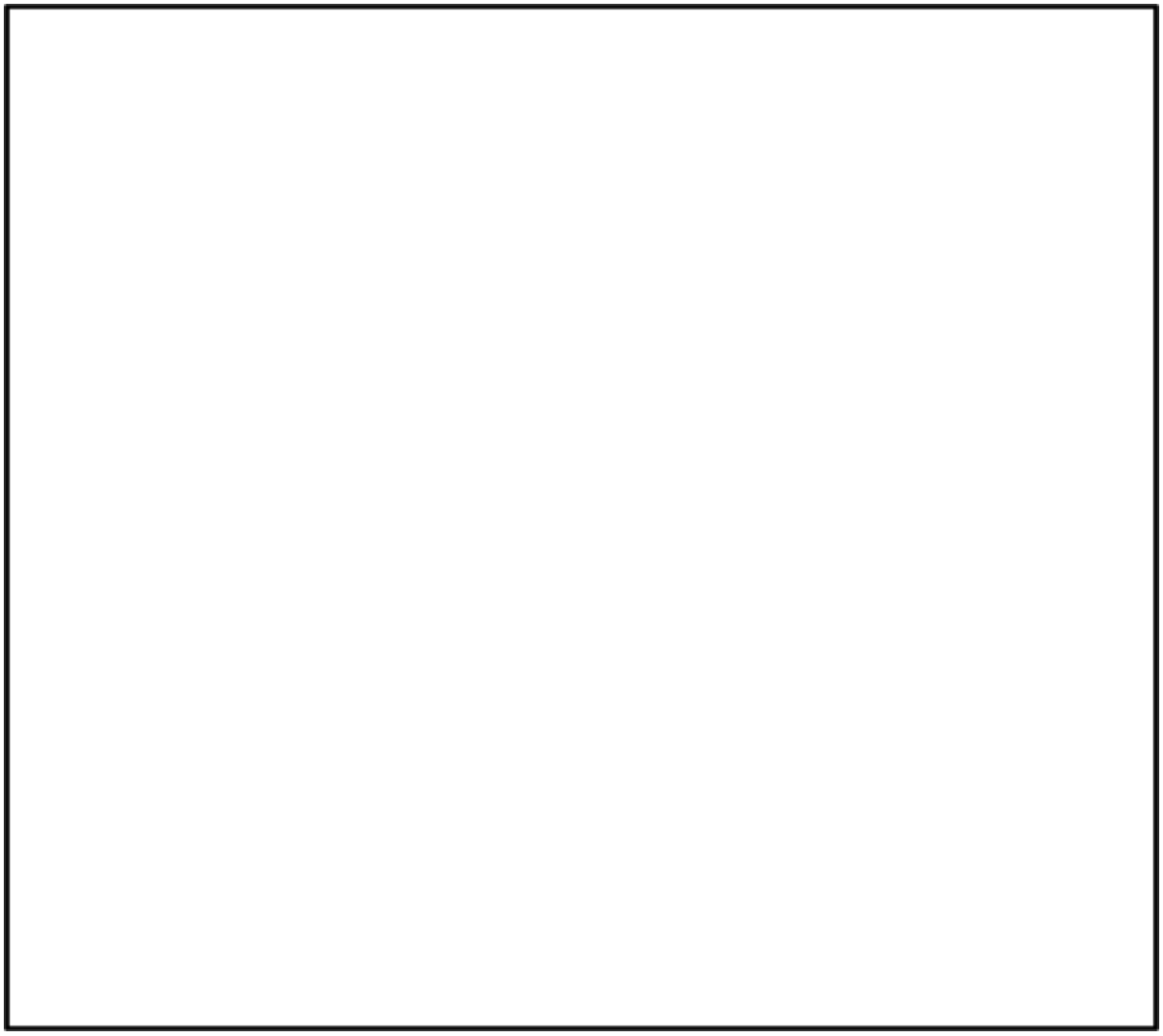
total score

9) A square, 2D array (rectangular, with the same number of rows and columns) is **tri-diagonal** if the only non-zero entries lie on the main diagonal (upper left to lower right) or on the diagonals immediately adjacent to the main diagonal. Below, the left and center arrays are tri-diagonal. The right array is not (the underlined 1,2 break the tri-diagonal definition). Note that zeros in the main and adjacent diagonals are acceptable (See left matrix)

1 2 0 0 0	1 2 0 0 0 0	1 2 0 **2**
1 2 3 0 0	4 2 3 0 0 0	2 1 1 **1**
0 0 2 3 0	0 1 2 4 0 0	0 1 1 1
0 0 1 1 0	0 0 1 1 1 0	0 0 1 1
0 0 0 1 1	0 0 0 2 3 4	
	0 0 0 0 1 2	

Write the method **triDiagonal()** that takes as input a square 2d array and returns true if and only if the array is tri-diagonal. Assume that the input array is square, fully initialized and row-major, like the arrays in question 6. (30 points)

Lehigh Username (Write in upper case letters): | | | | | | | @lehigh.edu



Lehigh Username (Write in upper case letters): | | | | | | | @lehigh.edu

total score

10) Two arrays are said to have the "**same content"** if all entries in the first array are in the second array and vice versa. Write a method **contains(a,b)** that returns true if and only if all entries in **b** are in array **a**. For example, for arrays **int x[]={1,2,3}, y[]={2,3}, z[]={2,1,3,1}**, contains(x,y) is true, while contains(y,x) is false. Note that if contains(a,b) and contains(b,a) are both true, then arrays a and b have the same content, based on the definition above. Next, write the method **sameContent()** that uses two calls to contains() to evaluate if two arrays have the same content. Thus, sameContent(x,z) should return true, while sameContent(x,y) should return false. (20 points)

write contains() here:

write sameContent() here: