Name (print): ________________________________

- **INSTRUCTIONS:**
  
  - Keep your eyes on your own paper and do your best to prevent anyone else from seeing your work.
  - Do NOT communicate with anyone other than the professor/proctor for ANY reason in ANY language in ANY manner.
  - This exam is closed notes, closed books, and no calculator.
  - Turn all mobile devices off and put them away now. You cannot have them on your desk.
  - Write neatly and clearly indicate your answers. What I cannot read, I will assume to be incorrect.
  - Stop writing when told to do so at the end of the exam. I will take 5 points off your exam if I have to tell you multiple times.
  - Academic misconduct will not be tolerated. Suspected academic misconduct will be immediately referred to the Emory Honor Council. Penalties for misconduct will be a zero on this exam, an F grade in the course, and/or other disciplinary action that may be applied by the Emory Honor Council.

- **TIME:** This exam has 6 questions on 14 pages including the title page. Please check to make sure all pages are included. You will have 50 minutes to complete this exam.

```
I commit to uphold the ideals of honor and integrity by refusing to betray the trust bestowed upon me as a member of the Emory community. I have also read and understand the requirements and policies outlined above.
```

Signature: ________________________________

<table>
<thead>
<tr>
<th>Question</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Points:</td>
<td>10</td>
<td>14</td>
<td>18</td>
<td>18</td>
<td>20</td>
<td>20</td>
<td>100</td>
</tr>
<tr>
<td>Score:</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
1. (10 points) For each entry below, state whether the code is correct, or has an error. If there is no error, write the output. If there is, show where the error is (you can circle the incorrect code) and explain why it is an error.

The following variables were initialized for you:

String s1="exam2";
int[] arr1={1, 2, 3, 4, 5};
String[] arr_s={"CS", "170"};

(a) System.out.println(arr1[arr1.length]);

Solution: ERROR: index out of range

(b) System.out.println(arr1[s1.length]);

Solution: ERROR: to get the length of a string, use string method s1.length()

(c) System.out.println(arr1[s1.length()]);

Solution: ERROR: index out of range

(d) System.out.println(arr1[s1.length()-1]);

Solution: 5

(e) System.out.println(s1.charAt(arr1[3]));

Solution: 2

(f) System.out.println( s1.charAt(arr_s.length) );

Solution: a

(g) System.out.println( arr1[ (int) s1.charAt(4) ] );

Solution: ERROR: s1.charAt(4) is "2", casting will change it to 50, not 2; out of range
(h) `System.out.println( arr_s[1].charAt(arr1[ s1.length()- 3 ])) );`

**Solution:** ERROR: `arr_s[1]` is "170", `charAt(3)` is out of range

**Solution:** Grading:
1-6, 1pt for each;
7,8, 2pt for each.
2. (14 points) Draw the array $A$ that would result after the following code is executed.

(a) ```java
char[] B = {'c', 's', '1', '7', '0'};
char[] A = new char[B.length];
for (int i = B.length -1; i >= 0; i--){
    A[i] = B[i];
}
```

(b) ```java
int[] A = { 1, 2, 3, 5, 8, 13};
for( int i=0; i < A.length-1; i++){
    int temp=A[0];
    A[i]=A[i+1];
}
```

(c) ```java
int[] A = { 1, 2, 3, 5, 8, 13};
for( int i=0; i < A.length-1; i++){
    int temp=A[i];
    A[i+1]=temp;
    A[i]=A[i+1];
}
```

<p>| | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>c</td>
<td>s</td>
<td>1</td>
<td>7</td>
<td>0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Solution: Grading:
part 1: 4 points;
part 2: 5 points;
part 3: 5 points.
3. (18 points) Read the following Java code. Then answer the questions.

```java
public class Q3 {
    public static double minScore = 60;

    public static double curve10Sqrt(double score) {
        double res = Math.sqrt(score) * 10;
        return res;
    }

    public static void curveAddPoint(double score, double addPoint) {
        double res = score + addPoint;
    }

    public static void curveAll(double[] score) {
        double[] res = score;
        for (int i = 0; i < res.length; i++) {
            res[i] += 10;
        }
    }

    public static boolean isFailed(double score) {
        return score < minScore;
    }

    public static void main(String[] args) {
        double[] myRawScore = {38, 47, 59};
        if (isFailed(myRawScore[0])) {
            System.out.println("Failed!");
        }

        double myNewScore1 = curve10Sqrt(myRawScore[0]);
        System.out.print("Score 1 After curving (square root times 10): ");
        if (isFailed(myNewScore1)) {
            System.out.println("Failed!");
        } else {
            System.out.println("Passed!");
        }

        curveAddPoint(myRawScore[1], 30);
        System.out.print("Score 2 After curving (adding 30 points): ");
        if (isFailed(myRawScore[1])) {
            System.out.println("Failed!");
        } else {
```
System.out.println("Passed!");
}

curveAll(myRawScore);
System.out.print("Score 3 After curving: ");
if(isFailed(myRawScore[2])){
    System.out.println("Failed!");
}else{
    System.out.println("Passed!");
}

(a) (3 points) Write one example for each type of parameter. Write down variable name and its data type
Class variable:

**Solution:** minScore; double

Local variable:

**Solution:** myRawScore; double array (or double[])

Parameter variable:

**Solution:** score in curve10Sqrt; double

(b) (1 points) What is the return datatype of method curve10Sqrt?

**Solution:** double

(c) (1 points) What is the input datatype(s) of method curveAddPoint?

**Solution:** double; double (both are double)

(d) (2 points) What does the method isFailed do?

**Solution:** It compares the input score with the class variable minScore; It returns true if score is lower than minScore. Otherwise it returns false.
(e) (11 points) What is the output of this program?

HINT:

* Math.sqrt() will return the square root of one value.
* Math.sqrt(38) is larger than 6, but smaller than 7.

Solution:

Failed!
Score 1 After curving (square root times 10): Passed!
Score 2 After curving (adding 30 points): Failed!
Score 3 After curving: Passed!

Grading:

First message, 1 point;
Score 1 to Score 3, 3 points for each
4. (18 points) Consider the following array \{3, 1, 9, 2, 10, 4, 8\}

(a) (3 points) Write out the values in the list that will be inspected during a linear search for the value 10.

Solution: \{3, 1, 9, 2, 10\}

(b) (6 points) The 2 sets of list below show the sequence of a list of numbers when it is sorted using one of the sorting algorithms we studied. Steps that have no changes to the arrays are being omitted. For each set of lists, label the name of the sorting algorithm after each set.

\begin{align*}
\text{Step 1: } & \{1, 3, 9, 2, 10, 4, 8\} \\
\text{Step 2: } & \{1, 2, 9, 3, 10, 4, 8\} \\
\text{Step 3: } & \{1, 2, 3, 9, 10, 4, 8\} \\
\text{Step 4: } & \{1, 2, 3, 4, 10, 9, 8\} \\
\text{Step 5: } & \{1, 2, 3, 4, 8, 9, 10\}
\end{align*}

Sorting algorithm _________________

Solution: SELECTION SORT

\begin{align*}
\text{Step 1: } & \{1, 3, 9, 2, 10, 4, 8\} \\
\text{Step 2: } & \{1, 2, 3, 9, 10, 4, 8\} \\
\text{Step 3: } & \{1, 2, 3, 4, 9, 10, 8\} \\
\text{Step 4: } & \{1, 2, 3, 4, 8, 9, 10\}
\end{align*}

Sorting algorithm _________________

Solution: INSERTION SORT

(c) (5 points) Now we can apply binary search for a value on the current sorted array \{1, 2, 3, 4, 8, 9, 10\}

Please write out the values that will be inspected during a binary search for the key 7.

Solution: 4, 9, 8

(d) (4 points) What is the return value of searching for key 7?

Solution: -5
5. (20 points) You’re helping Emory University to maintain OPUS on the exam score reporting system. Suppose for each department, the exam scores for each students are represented in a 2D-array:

- each row is the exam scores for one student.
- each column is the exam score for one course.

For example, if we have a 2D array, showing exam scores for 3 students (0,1,2), in 3 courses(0,1,2):

```java
double[][] scores =
{ {100,100,100},
  {90,90,90},
  {80,80,80} };
```

So, `score[1][1]` is the exam score of student 1 on course 1: 90 You can design your own user-defined method. Make sure the input and return data type are correct.

(a) Your job is to evaluate the AVERAGE EXAM SCORE for EACH STUDENT:

Write a user-defined method named `arrayRowMean`, which takes an 2D-array of `double` (Assume the 2D-array is not ragged), `array`. This method will return a new double array. Each element of this returned array is the mean value (i.e. average) of each ROW.

```java
double[] a = arrayRowMean(arr); // a is { 100, 90, 80 }
```

// ******** YOUR CODES HERE *******

**Solution:**
public static double arrayMean(double[] array) {
    double mean = 0;
    for (int i = 0; i < array.length; i++) {
        mean += array[i];
    }
    mean = mean / array.length;
    return mean;
}

public static double[] arrayRowMean(double[][] array) {
    int n = array.length;
    double[] res = new double[n];
    for (int i = 0; i < n; i++) {
        res[i] = arrayMean(array[i]);
    }
    return res;
}

Grading:
+1: function header (name, parameters, return type)
+2: variable initialization to correctly execute math
+2: uses loop correctly to examine all elements
+3: correctly get mean value for each row
+2: returns array correctly

(b) You'll get extra stipend if you can help to evaluate the AVERAGE SCORE for EACH COURSE:
write another user-defined method, named arrayColMean, which takes an 2D-array of double( Assume the 2D-array is not ragged), array. This method will return a new double array. Each element of this returned array is the mean value (i.e. average) of each COLUMN.
double[] b = arrayColMean(arr); // b is { 90, 90, 90 }
// ******** YOUR CODES HERE *******
Solution:

```java
public static double[] arrayColMean(double[][] array){
    int nRow = array.length;
    int nCol = array[0].length;
    double[] res = new double[nCol];
    for(int i = 0; i<nCol; i++){
        for(int j = 0; j<nRow; j++){
            res[i] += array[j][i];
        }
        res[i]=res[i]/nRow;
    }
    return res;
}
```

Grading:
+1: function header (name, parameters, return type)
+2: variable initialization to correctly execute math
+4: uses loop correctly to examine all columns; or correctly transpose matrix, then use rowMean method
+2: correctly get mean value for each column
+1: returns array correctly
6. (20 points) You wake up and find yourself back in the 80's. Two guys, Steve and Ronald, ask you to join their team for obvious reasons - you know computer science. They promise you a 1% stock share if you can help them to design a computer game “Rock-Scissors-Paper” for their "new-generation personal computer", Macintosh...

Now write a Java program for a simple game:

- 3 round of "Rock-Scissors-Paper” between you and computer
- Each round, computer will generate a hand shape randomly;
- User will input a hand shape by choosing 1: Rock, 2: Scissors, or 3: Paper.
- In each round, you can get 1 point ONLY when you WIN. If it is DRAW or LOSE, computer get one point.
- After each round, program will show:
  (1) Game result for each round: You win! or You Lose! (Again, draw shows lose too)
  (2) The current scores for both computer and user.
- At the end of the game, show final scores of computer and user.

**Hint:**

* Use Math.random() to generate random numbers. Make sure to convert it to 3 different integers.

* In order to avoid complicated logical expression to judge who won, use a 2D boolean array to show pair-wise relationship for any two hand shape. (e.g. Each row means hand shape of human; each column means hand shape of computer, then the value will represent in this case, whether human win or human lose).

* Part of the user-defined methods are given. Please use them in your main method. Make sure the input and return data type are correct.

**Sample Output:** when playing with your game, the output should be in the following format (result may vary, because it is generated randomly)

```
Welcome to play ROCK-SCISSOR-PAPER!!
----------------------------------------
Round 1
Choose between (1)Rock, (2)Scissors or (3)Paper: (input 1~3)3
 |-Computer choose: Paper
 |-you choose: Paper
You Lose!
```
Score: Computer(1), Human(0)
----------------------------------------
Round 2
Choose between (1) Rock, (2) Scissors or (3) Paper: (input 1~3)
 |-Computer choose: Rock
 |-You choose: Scissor
You Lose!
Score: Computer(2), Human(0)
----------------------------------------
Round 3
Choose between (1) Rock, (2) Scissors or (3) Paper: (input 1~3)
 |-Computer choose: Scissor
 |-You choose: Rock
You Win!
Score: Computer(2), Human(1)
----------------------------------------
Final Score:
Computer v.s. Human
2 : 1

import java.util.Scanner;
public class Q6{

    // 2d array for win/lose
    public static boolean[][] isWin=
    {
        //  ********** YOUR CODES HERE  **********
    };   

    public static void gameInfo(int humanHand, int computerHand){
        String[] hand = {"Rock", "Scissor", "Paper"};
        System.out.println(" |-Computer choose: "+hand[computerHand]);
        System.out.println(" |-You choose: "+ hand[humanHand]);
    }

    public static boolean playOneRound(int humanHand){
        // USE Math.random() to generate 3 random numbers
int computerHand;
// ******* YOUR CODES HERE *******

gameInfo(humanHand, computerHand);
boolean res = isWin[humanHand][computerHand];
return res;
}

public static void main(String[] args){
    int roundNumber = 3;
    int computerPoints = 0;
    int humanPoints = 0;

    System.out.println("Welcome to play ROCK-SCISSOR-PAPER!!");
    System.out.println("----------------------------------------");

    // ******* YOUR CODES HERE *******
import java.util.Scanner;
public class Q6{

//2d array for win/lose
public static boolean[][] isWin=
{false, true, false},
{false, false, true},
{true, false, false};

public static void gameInfo(int humanHand, int computerHand){
    String[] hand = {"Rock", "Scissor", "Paper"};
    System.out.println(" |-Computer choose: "+hand[computerHand]);
    System.out.println(" |-You choose: "+ hand[humanHand]);
}

public static boolean playOneRound(int humanHand){
    int computerHand = (int)(Math.random()*3);
    gameInfo(humanHand, computerHand);
    boolean res = isWin[humanHand][computerHand];
    return res;
}

public static void main(String[] args){
    int roundNumber = 3;
    int computerPoints = 0;
    int humanPoints = 0;

    System.out.println("Welcome to play ROCK-SCISSOR-PAPER!!");
    System.out.println("----------------------------------------");
    for (int r = 1; r <= roundNumber; r++){
        System.out.println("Round "+r);
        Scanner in = new Scanner(System.in);
        System.out.print("Choose between (1)Rock, (2)Scissors or (3)Paper: (input 1-3) ");
        int userInput = in.nextInt();
        if( playOneRound(userInput-1 ) ){
            humanPoints++;
            System.out.println("You Win!");
        }else{
        }else{
    }}}}
```java
computerPoints++;
    System.out.println("You Lose!");
}
System.out.println("Score: Computer("+computerPoints+"),
    Human("+humanPoints+"));
    System.out.println("----------------------------------------");
System.out.println("Final Score:");
System.out.println("Computer v.s. Human");
System.out.println(""+computerPoints+" \t\t"+humanPoints);
```

Grading:
+1 method header (name, parameters, return type)
+4 complete 2D-array to show win-lose relationship (or other logic structure)
+3 match the correct hand shape with parameter variable, local variable, index, etc.
+2 write the random number generating statement correctly
+3 uses loop to control the round of game
+3 calculate points of computer and human correctly
+4 print out game information correctly for each round and at the end of game