CSE 131 Spring 2019 — Exam 1 — Due at end of session

This exam is: **closed-book**, **NO electronic devices allowed**, and **closed-notes**. The exception is the "sage page" of the designated size on which you may have notes to consult during the exam.

Be sure you: **Provide legible answers in designated areas (credit will not be given for work that is difficult to read or not where expected)**, **Ensure you clearly fill in a single circle on multiple choice questions**, **Use indentation of your code to show its structure (but don't dwell on exact punctuation/syntax)**, **Leave the exam stapled together in its original order**, **Do *NOT* attach any other pages to the exam**. You are welcome to use the blank space on the exam for any scratch work.

If you need to leave the room for any reason prior to turning in your exam, you must leave your exam and any electronic devices with a proctor. **We do not clarify or explain anything during the exam session. State your assumptions if something is unclear and do the best you can.**

You must complete all the identifying information below correctly. Failure to do so is grounds for a zero on this exam:

1. Name (**print clearly**): ______________________________
2. Student ID (**print clearly; 1 digit per underline**): ___ ___ ___ ___ ___ ___
3. Which time do you typically attend studio (fill one)

   ○ 11:30—1:00 ○ 1:00—2:30 ○ 2:30—4:00 ○ 4:00—5:30
4. Which Urbauer lab do you typically sit in (best guess; fill one)

   ○ 216 ○ 218 ○ 222
5. You must sign the pledge below for your exam to count. The penalty for cheating will be decided during academic integrity review, but the instructors will recommend an F in this course as the minimum penalty.

   I have read the instructions on this page and I will neither give nor receive any unauthorized aid on this exam.

   ____

   (Sign above)

⟹ Do not proceed until told to do so! ⟸

⟹ Initial the top right corner of each page before starting ⟸

1. (14 points) For each expression below indicate the type and value of the result. Use quotation marks to indicate strings. The first row has been completed as an example.

Expression	Result Type	Result Value
"Good "+"Luck!"	○ boolean ○ double ○ int [✓ String]	"Good Luck!"
"1"+3+1	○ boolean ○ double ○ int ○ String	
1.0/2	○ boolean ○ double ○ int ○ String	
(!true) && false	○ boolean ○ double ○ int ○ String	
(1/2)<1	○ boolean ○ double ○ int ○ String	
1+(2+5)+"4"	○ boolean ○ double ○ int ○ String	
15%4	○ boolean ○ double ○ int ○ String	
true && (4<3)	○ boolean ○ double ○ int ○ String	

2. (4 points) Indicate the most appropriate data type for each concept:

(1) The name of your TA:
○ boolean ○ double ○ int ○ String

(2) Is your TA a CSE major?:
○ boolean ○ double ○ int ○ String

(3) The number of extensions you have completed:
○ boolean ○ double ○ int ○ String

(4) The probability of a snow in March:
○ boolean ○ double ○ int ○ String

3. (4 points) Given the following Java statement:

```
a = 1.0 / 2.0 + 3.5 * 7.3;
```

(1) Which operation is completed first?

○ = ○ / ○ + ○ *

(2) Which operation is completed second?

○ = ○ / ○ + ○ *

(3) Which operation is completed third?

○ = ○ / ○ + ○ *

(4) Which operation is completed last?

○ = ○ / ○ + ○ *

4. (8 points) Fill in the circle(s) whose `while`-loop code produces the same output as the following `for`-loop code.

This code using a `for` loop:

```
for(int i=0; i<12; i++) {
   System.out.println(i);
}
```

○ produces the same output as this code:

```
int i=0;
while (i<12) {
   System.out.println(i);
   i++;
}
```

○ produces the same output as this code:

```
int i=0;
while (i<12) {
   i++;
   System.out.println(i);
}
```

○ produces the same output as this code:

```
int i=0;
while (i<12) {
   System.out.println(i);
}
i++;
```

○ produces the same output as this code:

```
int i=0;
while (i<12) {
   System.out.println(i);
}
```

5. True or False and multiple choice. Fill in the correct circle:

   (1) (8 points) Consider the following loop:

   ```
   while(x < 4) {
       // some code
   }
   ```

   Which of the following <u>must</u> be true if the `while` loop ever exits? (fill in all that apply)

   ○ $x = 3$ ○ $x < 4$
   ○ $x \geq 4$ ○ $x = 5$

   (2) (6 points) Consider the following code:

   ```
   double x = Math.random();
   int    y = (int)(x*6 + 1);
   ```

   Which of the following best characterizes the possible values for `y`?

   ○ $0 \leq y \leq 6$ ○ $0 \leq y \leq 7$ ○ $1 \leq y \leq 6$ ○ $1 \leq y \leq 7$

   (3) (2 points) Including one loop within another loop is an example of:

   ○ Hyperlooping ○ Loopification
   ○ Nesting ○ Reiterating

   (4) (4 points) Given the following Java statement:

   ```
   int[][] array = new int[5][10];
   ```

   array.length will be:

   ○ 4 ○ 5 ○ 6
   ○ 9 ○ 10 ○ 11
   ○ 36 ○ 50 ○ 66

   (5) (5 points) Given the following Java statement:

   ```
   int[] array = new int[10];
   ```

   i. How many values can be stored in the array?

   ○ 8 ○ 9 ○ 10 ○ 11

   ii. What are valid indices for the array (ranges are inclusive)?

   ○ [0–8] ○ [0–9] ○ [0–10] ○ [1–8] ○ [1–9] ○ [1–10] ○ [1–11]

6. Show the output of each snippet of code in the area provided. Be as accurate as possible, including spacing and line usage. All output should start on the first dotted line on the right side of the paper (the leftmost dot corresponds to the left side of the console window). *The first line of the first problem has been filled in as an example.*

(1) (10 points)

```
for(int i=0; i < 3; i++) {
      if(i/2 == 0) {
         System.out.println(i);
      }
}
```

```
0
```

---

(2) (10 points)

```
for(int i=0; i < 3; i++) {
      for(int j=4; j>2; j--) {
         System.out.println("i:"+i+" j:"+j);
      }
}
```

---

(3) (10 points)

```
for (int i=0; i <= 3; ++i) {
    for (int j=0; j < i; ++j) {
       System.out.println(i + " " + j);
    }
}
```

7. Consider a two-dimensional square `boolean` array `A` such as the 7×7 one shown below. We say the array is square because it has the same number of columns and rows. To avoid clutter, cells whose values are `true` are shown with a t, and cells whose values are `false` are shown as blank.

`boolean[][] A`	column						
	0	1	2	3	4	5	6
row 0	t						
row 1			t				
row 2					t		
row 3							t
row 4							
row 5							
row 6							

In the above array, only the following cells have the value `true`:

`A[0][0], A[1][2], A[2][4], A[3][6]`

All other cells contain the value `false`.

Use the above information to go over the array and make sure you understand that the first index is the row and the second index is the column.

Following the instructions below, you will write code that determines some properties of an array `A` such as the one shown above. It is important to note that:

- The array above is 7×7, but any square array could be provided to you. Thus, `A` might be 3×3, or 8×8, or $N \times N$ array for positive (greater than zero) of N.
- The values in `A` shown above are just examples. The array you are provided has a value in each cell (`true` or `false`), and each cell could be either of those values.
- The problems below use the array shown above to illustrate what you are asked to do. However, you must keep in mind that the array above is just one possible example of an array `A` that is provided in the code.

(1) (5 points) Complete the code below so that each cell of the array `A` is randomly set to the value `true` or `false`. Your code must set the values independently, a word which here means that we cannot predict the value of one cell from any other cell's value. Recall that `Math.random()` returns an unpredictable `double` value from 0.0 to almost 1.0, and that each call to `Math.random()` returns an value that is independent, as far as we can tell, from all other calls.

Add code as needed in the blank lines below so that the code works properly, filling in the array with random boolean values and printing at the end how many of the cells are `true`.

```
public static void main(String[] args) {
  ArgsProcessor ap = new ArgsProcessor(args);
  int N = ap.nextInt("How many rows and columns?");
  boolean[][] A = new boolean[N][N];
  int numTrue = 0;   // at end, how many cells have value true?

..............................................................................
..............................................................................
..............................................................................
..............................................................................
..............................................................................
..............................................................................
..............................................................................
..............................................................................
..............................................................................
..............................................................................
..............................................................................

   System.out.println("The number of true cells in");
   System.out.println("  the array I just filled is");

..............................................................................
}
```

(2) (5 points) From here on, let's assume that every cell of the array `A` has some value, `true` or `false`. For this part of the problem, complete the code below so that the program prints the right message depending whether some column of the specified row `r` has the value `true`.

In the example shown at the beginning of this problem, row 2 has some column's value `true` (and so do rows 0, 1, and 3), but row 5 has no column's value `true` (and neither do rows 4 and 6).

```
public static void main(String[] args) {
  ArgsProcessor ap = new ArgsProcessor(args);
  int N = ap.nextInt("How many rows and columns?");
  boolean[][] A = new boolean[N][N];
  //
  // Some code such as what you wrote above that
  //   sets all the values of the
  //   array to true or false, randomly and independently
  //
  int r = ap.nextInt("Which row do you want to examine?");
  boolean found = false;

.....................................................................................
.....................................................................................
.....................................................................................
.....................................................................................
.....................................................................................
.....................................................................................
.....................................................................................
.....................................................................................

    if (found)
       System.out.println("There is a true value in row " + r);
    else
       System.out.println("There are only false values in row " + r);

}
```

(3) (5 points) Now we are interested in whether the diagonal containing a given cell has any value `true`.

boolean[][] A	0	1	2	3	4	5	6
row 0	t		☺				
row 1			t	☺			
row 2					t☺		
row 3						☺	t
row 4							☺
row 5							
row 6							

r	c	`true` value found on diagonal?
0	0	Yes
0	1	Yes
1	0	No
2	2	Yes
2	3	Yes
2	5	Yes
3	1	No
0	4	No
6	6	Yes

The above array is meant to be identical to the one at the beginning of this problem, except that the digonal containing cell `A[1][3]` is shown with smiley faces to illustrate the following example. Your code below should report a `true` value found if `r` and `c` are 1 and 3, respectively because of the `true` value present at `A[2][4]`. Other results for this example are shown to the right of the array above.

As indicated by the example diagonal of smiley faces

- a diagonal always runs from upper-left to lower-right
- the cells of a diagonal always touch at their corners
- big hint: For every cell on a given diagonal, the difference between its row coordinate and its column coordinate is the same constant.

Let's investigate the big hint. Consider every cell on the smiley face diagonal: `A[0][2]`, `A[1][3]`, `A[2][4]`, `A[3][5]`, and `A[4][6]`. The difference of of row and column coordinates for each cell is -2. Also, no other cell's row and column difference is -2.

The code you complete on the next page prompts the user for a row `r` and column `c`. The code you write on the blank lines must then set the boolean variable `found` `true` if any cell on the diagonal with `A[r][c]` has the value `true`. The actual cell from prompting the user, `A[r][c]`, may have value `true` or `false`.

```
public static void main(String[] args) {
  ArgsProcessor ap = new ArgsProcessor(args);
  int N = ap.nextInt("How many rows and columns?");
  boolean[][] A = new boolean[N][N];
  //
  // Some code such as what you wrote above that
  //   sets all the values of the
  //   array to true or false, randomly and independently
  //
  int r = ap.nextInt("Which row do you want to examine?");
  int c = ap.nextInt("And which column?");
  boolean found = false;


..........................................................................
..........................................................................
..........................................................................
..........................................................................
..........................................................................
..........................................................................
..........................................................................
..........................................................................
..........................................................................
..........................................................................
..........................................................................
..........................................................................
..........................................................................
   if (found) {
      System.out.println("There is a true value");
      System.out.println("on the diagonal with row " + r);
      System.out.println("and column " + c);
   }
   else {
      System.out.println("There are only false values");
      System.out.println("on the diagonal with row " + r);
      System.out.println("and column " + c);
   }
}
```