COS126 **Written Exam 1** **Spring** '22

Instructions. This exam has seven (7) questions worth a total of seventy (70) points. You have fifty (50) minutes.

This exam is preprocessed by computer. Write neatly and legibly. If you use a pencil, write darkly. Write all answers inside the designated rectangles and nothing else (e.g. no scratch work inside designated rectangles). Fill in circles completely: ⚫ (not ✔ or ✘). If you change your mind, you must erase completely and fill in another circle!

Resources. The exam is closed book, except that you are allowed to use a one-page reference sheet (8.5-by-11 paper, one side, in your own handwriting). No electronic devices are permitted.

Discussing this exam. Discussing the contents of this exam before solutions have been posted is a violation of the Honor Code.

This exam. Do not remove this exam paper from this room. Print your name, NetID, precept, and the room in which you are taking the exam in the space below. Also, **copy and sign the Honor Code pledge**. You may enter this information now. Again, please write neatly and legibly.

NAME:

NETID (not email alias):

PRECEPT:

EXAM ROOM: ◯ McCosh 50 ◯ McDonnell A02 ◯ CS 104 ◯ CS 301/302 ◯ OTHER _____

"I pledge my honor that I will not violate the Honor Code during this examination."

_____

_____

SIGNATURE: _____

Question 1	**Data Types**	**10** points

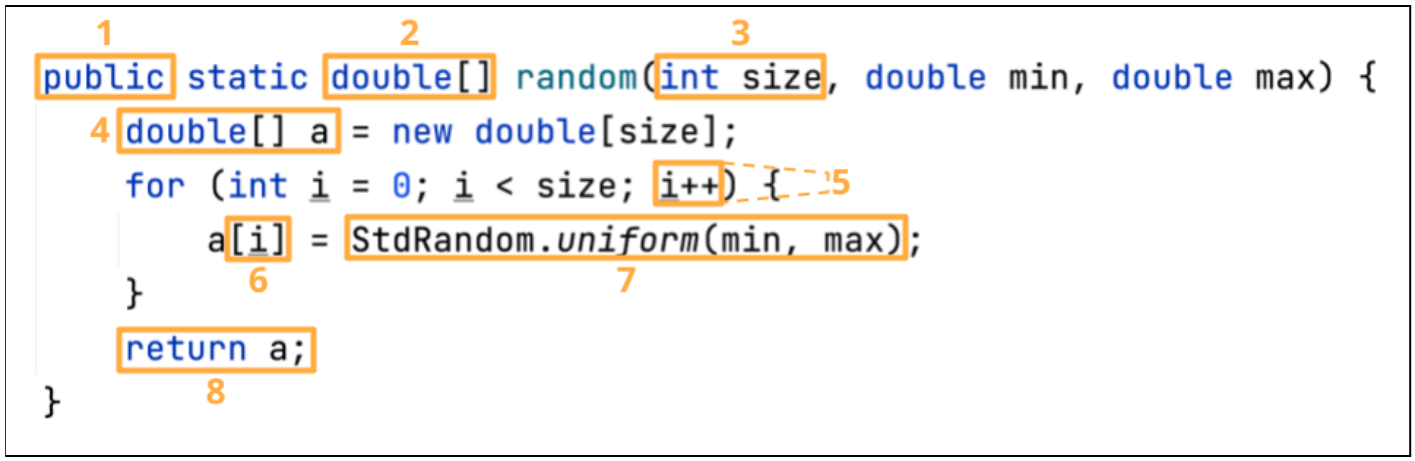
For each row, suppose that the expression shown on the left is the sole argument of a `StdOut.println()` statement. For an example, see row 1: what would be printed if `StdOut.println("1");` was executed?

Fill in exactly one circle corresponding to the output or select **ERROR** if a compile- or run-time error will occur. For the last row, note that the ASCII value for the lower-case `'a'` is 97.

Expression	**ERROR**	**true**	**false**	**0**	**0.0**	**0.5**	**1**	**1.0**	**1.5**	**2**	**2.0**	**2.5**
`"1"`	○	○	○	○	○	○	●	○	○	○	○	○
`3 / 2`	○	○	○	○	○	○	○	○	○	○	○	○
`0.5 * 2`	○	○	○	○	○	○	○	○	○	○	○	○
`(double) 5 % 2`	○	○	○	○	○	○	○	○	○	○	○	○
`Math.max(0,2,1)`	○	○	○	○	○	○	○	○	○	○	○	○
`2 + "." + 5`	○	○	○	○	○	○	○	○	○	○	○	○
`Integer.parseInt("2.5")`	○	○	○	○	○	○	○	○	○	○	○	○
`(int) "1.5"`	○	○	○	○	○	○	○	○	○	○	○	○
`3 >= 2 <= 4`	○	○	○	○	○	○	○	○	○	○	○	○
`!(1 <= 2) && (3 != 4)`	○	○	○	○	○	○	○	○	○	○	○	○
`(int) 'a' - 97.0`	○	○	○	○	○	○	○	○	○	○	○	○

Question 2	**Java Syntax and Terminology**	**10** points

In each row below, fill in exactly one circle for the letter corresponding to the best-matching description of that part of the program. You may use each letter once, more than once, or not all.



A – API definition

B – API function call

C – array index

D – conditional statement

E – transfers control to caller

F – function argument

G – function signature

H – increment statement

I – initialization statement

J – declaration statement

K – specifies the return type of the function

L – denotes that other programs can call this function

	A	B	C	D	E	F	G	H	I	J	K	L
1	○	○	○	○	○	○	○	○	○	○	○	○
2	○	○	○	○	○	○	○	○	○	○	○	○
3	○	○	○	○	○	○	○	○	○	○	○	○
4	○	○	○	○	○	○	○	○	○	○	○	○
5	○	○	○	○	○	○	○	○	○	○	○	○
6	○	○	○	○	○	○	○	○	○	○	○	○
7	○	○	○	○	○	○	○	○	○	○	○	○
8	○	○	○	○	○	○	○	○	○	○	○	○

Question 3	**Conditionals and Loops**	**10** points

Consider this incomplete program – with blanks **A**, **B**, and **C**:

```
1   public class Q3 {
2      public static void main(String[] args) {
3         int n =  __A__ ;
4         int sum = 0;
5         for (        B        ) {
6            if (        C        )
7               sum++;
8            else
9               sum--;
10        }
11          StdOut.println(sum);
12     }
13  }
```

For each row in the last table, suppose you complete the program by inserting the given value of `n` (first column) into blank **A** on line 3**,** the `for` loop header (second column; i.e. *<initialize>; <boolean expression>; <increment>*) into blank **B** on line 5 and the boolean expression (third column) into blank **C** on line 6.

In the last column, write the output of the program. If the program will never terminate, write **NONE**.

We recommend that you trace this program by creating and filling in a table that at least keeps track of the following: the value of **x**, the boolean expression on line 6 (**C**), and the change to **sum** that occurs for each iteration of the loop (you may want to keep track of more information).

For an example, see the tracing table on the right for row 1 in the below table.

x	**Value of C**	**Change to sum**
0	true	+1
1	false	-1
2	true	+1
		sum = 1

Value of n (A)	**Loop Header (B)**	**Boolean Expression (C)**	**Output**
3	`int x = 0; x < n; x++`	`x % 2 == 0`	**1**
4	`int x = 0; x < n; x++`	`x % 3 != 1`	
5	`int x = 1; x < n; x *= 2`	`x * 2 < n`	
4	`int x = 1; x % n != 1; x++`	`x % 4 == 0`	
12	`int x = 0; x < n; x *= 3`	`x % 2 == 0`	

Question 4	**Arrays**	**10** points

Part A

Determine whether each of the following lines of code triggers an error (i.e. a compile- or run-time error) or if it compiles and runs successfully (i.e., **NO ERROR**).

	ERROR	**NO ERROR**
`double[] a = new int[10];`	◯	◯
`int[3] b = new int[3];`	◯	◯
`int[][] c = {{1,2,3}, {4, 5, 6}, {7, 8, 9}};`	◯	◯
`int[] d = {1, 2, 3}; int x = d[3];`	◯	◯
`int[] e = {1, 2, 3}; int[] h = e;`	◯	◯
`int[][] f = new int[10];`	◯	◯

Part B

Consider the following array of length `N`, where the values `1`, `2`, `3` are repeated multiple times:

```
int[] arr = {1, 2, 3, 1, 2, 3, …, 1, 2, 3, 1, 2, 3};
```

Fill in the circle corresponding to the value for each expression. If evaluating the expression would produce a compile- or run-time error, fill in the circle labeled **`ERROR`**.

Expression	**ERROR**	**1**	**2**	**3**
`arr[2]`	◯	◯	◯	◯
`arr[N-3]`	◯	◯	◯	◯
`arr[N]`	◯	◯	◯	◯
`arr[-1]`	◯	◯	◯	◯
`arr[arr[N-2]]`	◯	◯	◯	◯

Question 5	**StdIn, StdOut, Redirection, Piping**	**10** points

Consider the following program:

```
1  public class Q5 {
2     public static void main(String[] args) {
3        int n = StdIn.readInt();
4        StdOut.println(n + 2);
5     }
6  }
```

Assume the following:

a. The file `in.txt` contains only a single number **2**.
b. Before running each command, assume that the file `out.txt` does not exist.
c. If keyboard input is needed, you will type the number **3** and press the `<Enter>` key.

Complete the table. **Y** denotes yes and **N** denotes no. For the **ERROR** column, if the command results in an error, select **Y** and leave the rest of the row blank; otherwise, select **N** and complete the rest of the row. For the second to last column about `out.txt`, write the output that will be redirected to `out.txt`; if `out.txt` is not created, write **NONE**. For the last column about printed output, write the output that will be printed to the terminal; if nothing will be printed write **NONE**. For brevity, we write "`java`" as a shorthand for "`java-introcs`".

Command	**ERROR**	Is keyboard input required?	What output will be redirected to `out.txt`? (write **NONE** if `out.txt` is not created)	What output (if any) will be printed? (write **NONE** if no output)
`java Q5`	◯ **Y** ◯ **N**	◯ **Y** ◯ **N**		
`java Q5 < in.txt`	◯ **Y** ◯ **N**	◯ **Y** ◯ **N**		
`in.txt > java Q5`	◯ **Y** ◯ **N**	◯ **Y** ◯ **N**		
`java Q5 | java Q5 | java Q5`	◯ **Y** ◯ **N**	◯ **Y** ◯ **N**		
`java Q5 < in.txt > out.txt`	◯ **Y** ◯ **N**	◯ **Y** ◯ **N**		
`java Q5 < in.txt > out.txt | java Q5`	◯ **Y** ◯ **N**	◯ **Y** ◯ **N**		
`java Q5 < in.txt | java Q5 > out.txt`	◯ **Y** ◯ **N**	◯ **Y** ◯ **N**		

Question 6	**Functions**	**10** points

Consider the following functions:

```
public static int f(int[] a) {
    int s = 0;
    for (int i = 0; i < a.length; i++) {
        if (i % 2 == 0) s += a[i];
        else s -= a[i];
    }
    return s;
}
```

```
public static int g(int[] a) {
    int t = a[0];
    a[0] = a[1];
    a[1] = t;
    return f(a);
}
```

```
public static int h(int[] a) {
    int[] b = new int[a.length];
    for (int i = 0; i < b.length; i++) {
        b[i] = a[i] * 2;
    }
    return f(b);
}
```

Assume that the arrays y and z are defined as follows:

```
int[] y = {1, 2, 3};
int[] z = {3, 5, 7};
```

Part A

After evaluating the following function calls, has array y been mutated (i.e. changed)?

f(y)	◯ **Y** ◯ **N**
g(y)	◯ **Y** ◯ **N**
h(y)	◯ **Y** ◯ **N**

Part B

Assume that the arrays y and z are reset to their original values before making each function call. What do the following functions return?

	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
f(y)	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯
f(z)	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯
g(y)	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯
g(z)	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯
h(y)	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯
h(z)	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯

Question 7	**Recursion**	**10** points

Consider the following function:

```
1   public static String f(int x) {
2       if (x == 0) return "" + 0;
3       if (x == 1) return f(x - 1) + 1;
4       return x + f(x - 2) + (x - 1);
5   }
```

Suppose you call the above function `f()` with various values of `x` given in the leftmost column of each row below. Fill in the circle corresponding to the `i`th character (**zero-indexed**, second column) of the return value for `f(x)` (e.g., the 0th character of `"123"` is `'1'` and the 2nd character is `'3'`). If the `i`th character does not exist, select **`ERROR`**. For example, for the first row, you should select the option that corresponds to the 0th character (zero-indexed) in the return value for `f(0)`. Off-by-one errors will **not** receive partial credit.

x	**i^{th} char in f(x)**	**ERROR**	**'0'**	**'1'**	**'2'**	**'3'**	**'4'**
0	0	◯	◯	◯	◯	◯	◯
1	1	◯	◯	◯	◯	◯	◯
2	0	◯	◯	◯	◯	◯	◯
3	1	◯	◯	◯	◯	◯	◯
4	4	◯	◯	◯	◯	◯	◯