COS 126 Written Exam 1 Spring 2020

There are seven questions on this exam, each worth 2.5 points. There is one question per lecture, numbered in order of the lectures, *not in order of difficulty*. If a question seems difficult to you, skip it and come back to it. You will have 50 minutes to complete the exam.

This exam is preprocessed by computer. You answer questions by filling in circles *completely* with a dark pencil. If you change your mind, you must erase *completely* and fill in another circle!

<div align="center">

Do this ● not this ✓ or this ✗ or this ✗.

</div>

Resources. You may reference your optional one-sided 8.5-by-11 handwritten "cheat sheet" during this exam. You may not use the textbook, your notes, or any electronic devices. You may not communicate with anyone except the course staff during this exam.

Discussing this exam. Due to travel for extracurriculars and sports, some of your peers will take this exam later. Do not discuss its contents with anyone who has not taken it.

This page. Do not remove this exam from the exam room. Fill in this page now, but do not start the exam until you are told.

Name

NetID

Precept

Exam Room

"I pledge my honor that I have not violated the Honor Code during this examination."

_____

_____

[*copy the pledge here*]

_____

[*signature*]

Q1. Types and Casts (2.5 points).

Suppose that the code at left in each row is the sole argument of a `StdOut.print()` statement. Fill in the circle corresponding to the string it would print.

	0	1	1.0	1.5	2	2.0	3.0
3/2	○	●	○	○	○	○	○
2/3	●	○	○	○	○	○	○
3 * 0.5	○	○	○	●	○	○	○
3/2 + 3/2	○	○	○	○	●	○	○
(int) 3 * 0.5	○	○	○	●	○	○	○
3 * (int) 0.5	●	○	○	○	○	○	○
(int) (3 * 0.5)	○	●	○	○	○	○	○
(3 / 2) * 2.0	○	○	○	○	○	●	○
(3 / 2.0) * 2	○	○	○	○	○	○	●
3/2 + "." + 2/3	○	○	●	○	○	○	○

Q2. Basic definitions (2.5 points).

In the box to the left of each description, write the letter corresponding to the Java language entity that best matches. A letter may be used once, more than once, or not at all, but there is only one correct answer for each box.

D Java data type that has only two values

F A `double` operator

F A `String` operator

C A `double` literal

I Source-code representation of π

B A set of values and operations on those values

E A Java library function

A Data structure for a sequence of values of the same type

I Java data type for integers

I A `boolean` operator

A   array

B   data type

C   3.14

D   boolean

E   Math.sqrt()

F   +

G   binary

H   StdOut

I   *none of the above*

Q3. Loops and conditionals (2.5 points).

Suppose that this code is in the file Q3.java

```
public class Q3
{
    public static void main(String[] args)
    {
        int N = Integer.parseInt(args[0]);
        String r = "+";
        for (int i = 1; i <= N; i++)
            if ((i % 2) == 0)
                r = r + i + r;
            else r = i + r + i;
        System.out.println(r);
    }
}
```

and that you compile it using the command javac-introcs Q3. Study the code to verify that the command java-introcs Q3 1 prints the string "1+1" and javac-introcs Q3 2 prints the string "1+121+1".

Then fill in the circle on each row corresponding to the specified character. You must fill in exactly one circle in each row.

	1	2	3	4	5
first character printed by java-introcs Q3 3	○	○	●	○	○
first character printed by java-introcs Q3 4	○	○	●	○	○
tenth character printed by java-introcs Q3 5	○	○	●	○	○
second character printed by java-introcs Q3 5	○	○	●	○	○
next-to-last character printed by java-introcs Q3 6	○	○	●	○	○

Q4. Arrays (2.5 points).

For each code snippet at left, fill in the *one* circle at right that describes whether it is legal or whether it will lead to an error.

	legal code	compile-time error	run-time error
`int[] a;`	●	○	○
`int[] b = 0;`	○	●	○
`int[] c = int[10];`	○	●	○
`int[] d = new int[10];`	●	○	○
`int[][] e = new int[10];`	○	●	○
`int[10] f = new int[10];`	○	●	○
`int[] g = new int g[10];`	○	●	○
`int[] h = {1, 2, 3}; int t = h;`	○	●	○
`int[] i = {1, 2, 3}; int[] u = i;`	●	○	○
`int[] j = {1, 2, 3}; int v = j[3];`	○	○	●

Q5. I/O (2.5 points).

Carefully study the following Java program.

```
public class Q5
{
    public static void main(String[] args)
    {
        int N = args.length;
        String[] a = new String[N * 2];
        for (int i = 0; i < N; i++)
        {
            a[i] = args[i];
            a[i + N] = args[N - i - 1];
        }
        for (int i = 0; i < a.length; i++)
            StdOut.print(a[i] + " ");
        StdOut.println();
    }
}
```

In the box to the left of each command, write the letter corresponding to the output it produces. A letter may be used once, more than once, or not at all.

A is is is is

E java-introcs Q5 now is

B is is now now

A java-introcs Q5 is is

C is now is now

D java-introcs Q5 is now

D is now now is

E now is is now

E java-introcs Q5 is now | java-introcs Q5 now is

F now is now is

G now now is is

H java-introcs Q5 is is | java-introcs Q5 now now

H now now now now

Q6. Methods (2.5 points).

Consider the following Java program, which has no purpose other than to test your understanding of methods, arguments, and scope.

```
public class Q6
{
    public static void f(int[] a, int k, int m)
    {
        double[] b = f(a, 2.6);
        a[0] = a[0] * m;
        a[2] = (int) (b[0] + b[1]);
        m = m + 5;
    }
    public static double[] f(int[] a, double x)
    {
        double[] b = new double[2];
        b[0] = a[0] + x;
        b[1] = a[1] + x;
        return b;
    }
    public static void main(String[] args)
    {
        int[] a = {2, 3, 4};
        int k = 5;
        int m = 5;
        f(a, k, m);
    }
}
```

Fill in the one circle on each row corresponding to the value of the given expression just *after* the call f(a, k, m) has returned control to main().

	0	1	2	3	4	5	6	7	8	9	10
k	○	○	○	○	○	●	○	○	○	○	○
m	○	○	○	○	○	●	○	○	○	○	○
a[0]	○	○	○	○	○	○	○	○	○	○	●
a[1]	○	○	○	●	○	○	○	○	○	○	○
a[2]	○	○	○	○	○	○	○	○	○	○	●

Q7. Recursion (2.5 points).

Consider the following recursive function:

```
public static int ack(int m, int n)
{
    if (m == 0) return n + 1;
    if (n == 0) return ack(m - 1, 1);
    return ack(m - 1, ack(m, n - 1));
}
```

Fill in the circle corresponding to the value of ack() for each given value of the arguments. You must fill in exactly one circle in each row.

	4	5	6	7	8	15	16	17	509	510	511
ack(0, 6)	○	○	○	●	○	○	○	○	○	○	○
ack(1, 3)	○	●	○	○	○	○	○	○	○	○	○
ack(1, 6)	○	○	○	○	●	○	○	○	○	○	○
ack(2, 6)	○	○	○	○	○	●	○	○	○	○	○
ack(3, 6)	○	○	○	○	○	○	○	○	●	○	○