COSC–111 Introduction to Computer Science I
Solution to Midterm 01

1) Provide short answers (at most two sentences) to each of the following questions

1.a) How do you start a comment in Java?

A: With a double slash: `//`.

1.b) What is the purpose of casting?

A: Casting allows to "convert" a value of one type to a value of a different type. Casting does not convert a *variable*, it converts a value, which could potentially be the value of a variable.

1.c) Is the following sentence correct:"There exists a program flow that can be implemented with a `for` loop but not with a `while` loop"? Please briefly explain your answer.

A: The sentence is not correct. For any while loop we can design an equivalent for loop and vice versa.

1.d) What is the difference between `4`, `4.0`, and `"4"` when they are on the right of an equal sign (i.e., =), and followed by a semicolon? For each of them, what could be on the left of the equal sign?

A: They are values of different types, respectively an `int`, a `double`, and a `String`. On the left you could find, for example, the following code:

```
int x = 4; // but also "double x = 4" is possible!
double y = 4.0;
String z = "4";
```

2) Each of the following code snippets either does not compile or it does not run correctly. Please explain the error and give a simple modification that would make the code compile and run correctly.

Id A)

```
double x = 0.2;
int y = int(x);
```

Id B)

```
z = 5;
double j = 2;
int z = 45;
j *= z;
```

Id C)

```
int x = 6;
while (x > 0) {
    x--;
   for (int x=5; x % 6 > 0; x--) {
      System.out.print("*");
    }
    System.out.println();
}
```

Id D)

```
for (double x = 10; (x * 3) % 2 == 0; ) {
      System.out.println(x);
}
```

A: (Many corrections are valid, the following ones are the easiest)

- A) The syntax for the casting is incorrect. It should be:

```
double x = 0.2;
int y = (int) x;
```

- B) The variable `z` is used before being declared. Note that assigning an int to a double is a perfectly valid instruction. Also, the operator `*=` is a valid operator similar to `+=`, `-=`, `/=`. The code should be

```
int z = 5;
double j = 2;
z = 45;
j *= z;
```

- C) The variable `x` is declared twice, once on the first line and then again in the declaration of the for loop. A possible solution is to rename one of the two variables, e.g., to `y`:

```
int x = 6;
while (x > 0) {
    x--;
    for (int y=5; y % 6 > 0; y--) {
      System.out.print("*");
    }
    System.out.println();
}
```

- D) The code causes an infinite loop because the variable `x` is never modified in the loop. As mentioned earlier, assigning an integer value to a double is a perfectly valid operation. A possible solution is to decrement it:

```
for (double x = 10; (x * 3) % 2 == 0; x--) {
      System.out.println(x);
}
```

  This modification will make the program go through only one iteration of the loop, to print only one line with content `10`.

3) Two of the three following code snippets have the same output. One does not.

3.a) What is the output of the two snippets with the same output?

3.b) Which snippet does not print the same output?

3.c) How can you minimally modify this snippet to have the same output as the other two?

Snippet Id	Code

A	`for (int i = 0; i < 10; ++i) {`
	`  if ((i + 1) % 3 != 1) {`
	`  System.out.print("-");`
	`  } else {`
	`  System.out.print("*");`
	`  }`
	`}`
	`System.out.println();`

B	`for (int x = 11; x > 0; x /= 3) {`
	`  System.out.print("*--");`
	`}`
	`System.out.println("*");`

C	`int j = 10;`
	`while (j > 0) {`
	`  j--;`
	`  if (j % 3 == 1) {`
	`  System.out.print("*");`
	`  } else {`
	`  System.out.print("-");`
	`  }`
	`}`
	`System.out.println();`

A: Snippets A and B print `*--*--*--*`. The snippet C prints `--*--*--*-`. To fix C, one can move the `j--;` line to be the last instruction inside the loop body. An alternative solution is to keep `j--` where it is but have `j` start at 11 and change the while condition to `j > 1`.

4) The library is implementing a system of fines for late returns. The fines will depend on the status of the patron (faculty/staff or student), and how late the book is returned: slightly (at most 7 days late), medium (between 8 and 14, included), and very (15 and over). Consider the following code:

```
int late = 0;
while (late <= 0) {
  System.out.print("How late (in days) is the return? ");
  late = keyboard.nextInt();
}
System.out.print("Is the patron a student? ");
boolean isStudent = keyboard.nextBoolean();
int fine = 2;
if (late <= 7 && ! isStudent || late > 7) {
   fine += 2;
}
if (late > 14 || 7 < late && !isStudent && 14 >= late) {
   fine += 2;
}
if (14 < late && !isStudent) {
   fine += 3;
}
System.out.println("The fine is: " + fine);
```

How much is the fine for faculty/staff and for students for slightly, medium, and very late returns? Draw and fill a table like the one below in your answer sheet.

How Late	Students	Faculty/Staff
Slightly	2	4
Medium	4	6
Very	6	9

5) Using loops (for and/or while) and if statements, write code that asks the user for a positive odd integer value (and keeps asking until these conditions are satisfied), stores the value in a variable named `size`, and then print a pattern like the following, depending on the value of `size`. Each pattern has exactly `size` lines and columns. Your code must work for any positive odd integer value of `size`.

`size`	1	3	5	7	...

Pattern	`+`	`---`
		`-+-`
		`---`
			`-----`
			`| | |`
			`--+--`
			`| | |`
			`-----`
				`-------`
				`| | |`
				`| | |`
				`---+---`
				`| | |`
				`| | |`
				`-------`	

A: Asking the user to keep typing a value until it satisfies some specific condition is how we introduced loops.

A way to think about how to structure is to see that the code should:

- print vertical signs on the first, middle, and last columns of the lines that are *not* the first, the middle, or the last lines;
- print `size` dashes on the first and last line;
- print some dashes, a plus sign, and some more dashes on the middle line.

So the idea is to iterate over the lines and over the columns, and decide what to print based on the "coordinate" (i.e., line and column) that we are currently at. One way to implement this code is the following:

```
import java.util.Scanner;
public class Window1 {
    static Scanner kb = new Scanner(System.in);
    public static void main(String[] args) {
        int size = 2;
        while (size % 2 == 0 || size <= 0) {
            System.out.print("Input an odd positive integer: ");
            size = kb.nextInt();
        }
        for (int rows = 0; rows < size; rows++) {
            for (int cols = 0; cols < size; cols++) {
                if (rows == 0 || rows == size - 1 || rows == size / 2) {
                    if (rows == size / 2 && cols == size / 2) {
                        System.out.print("+");
                    } else {
                        System.out.print("-");
                    }
                } else {
                    if (cols == 0 || cols == size - 1 || cols == size / 2) {
                        System.out.print("|");
                    } else {
                        System.out.print(" ");
                    }
                }
            }
            System.out.println();
        }
    }
}
```

Another possible implementation is the following:

```
import java.util.Scanner;
public class Window2 {
    static Scanner kb = new Scanner(System.in);
    public static void main(String[] args) {
        int size = 2;
        while (size % 2 == 0 || size <= 0) {
            System.out.print("Input an odd positive integer: ");
            size = kb.nextInt();
        }
        for (int rows = 0; rows < size; rows++) {
            if (rows == 0 || rows == size - 1 || rows == size / 2) {
                for (int cols = 0; cols < size; cols++) {
                    if (rows == size / 2 && cols == size / 2) {
                        System.out.print("+");
                    } else {
                        System.out.print("-");
                    }
                }
                System.out.println();
            } else {
                for (int cols = 0; cols < size; cols++) {
                    if (cols == 0 || cols == size -1 || cols == size / 2) {
                        System.out.print("|");
                    } else {
                        System.out.print(" ");
                    }
                }
                System.out.println();
            }
        }
    }
}
```