

**Northeastern Illinois University**  
**CS200-1, Programming I, Summer 2017**  
**Homework 7**  
**Due date: Thursday 7/20/2017 at 1:00 p.m.**

**Problem 1:**

Create a class named **Problem1**, and create a main method, inside the main method prompt the user to enter a positive integer **n**. The program must error check (keep prompting and reading integers until you get a positive integer **n**). The program then will create an integer array **arr** of size **n**, and fill up that array with random integers between 1 and 500.

- Create a method named **getMax**, which takes an integer array **arr** and returns an integer, the method should find the maximum value in the integer array **arr**.
- Create a method named **getMin**, which takes an integer array **arr** and returns an integer, the method should find the minimum value in the integer array **arr**.
- Create a method named **sumValues**, which takes an integer array **arr** and returns an integer, the method should find the sum of all values in the integer array **arr**.
- Create a method named **getAverage**, which takes an integer array **arr** and returns a double, the method should find the average of all values in the integer array **arr**.
- Create a method named **greaterThanAverage**, which takes an integer array **arr** and returns an integer, the method should find how many values were greater than average in the integer array **arr**.
- Create a method named **countInc**, which takes an integer array **arr** and returns an integer, the method should find how many times subsequent value increases (the next value is bigger) in the integer array **arr**.
- Create a method named **countDec**, which takes an integer array **arr** and returns an integer, the method should find how many times subsequent value decreases (the next value is smaller) in the integer array **arr**.
- Create a method named **printArray**, which takes an integer array **arr** and returns no value, the method prints the entire array **arr**.
- Create a method named **printArray3PerLn**, which takes an integer array **arr** and returns no value, the method prints the entire array **arr** with at most 3 elements per line.
- Make the proper calls from the main method to test all the methods listed above.

- Below are two sample runs.
- Copy the output to a text file named Problem1.txt

```

Enter an integer n, greater than 0:  -1
Enter an integer n, greater than 0:  8
Array on one line:
101 324 408 341 311 5 480 251
Maximum: 480
Minimum: 5
Sum: 2221
Average: 277.625
Number of integers greater than average: 5
Number of times subsequent value increases : 3
Number of times subsequent value decreases : 4
Array with 3 elements per line:
101 324 408
341 311 5
480 251

```

```

Enter an integer n, greater than 0:  -5
Enter an integer n, greater than 0:  0
Enter an integer n, greater than 0:  14
Array on one line:
2 443 156 135 427 475 205 293 345 356 166 118 315 306
Maximum: 475
Minimum: 2
Sum: 3742
Average: 267.2857142857143
Number of integers greater than average: 8
Number of times subsequent value increases : 7
Number of times subsequent value decreases : 6
Array with 3 elements per line:
2 443 156
135 427 475
205 293 345
356 166 118
315 306

```

## **Problem 2:**

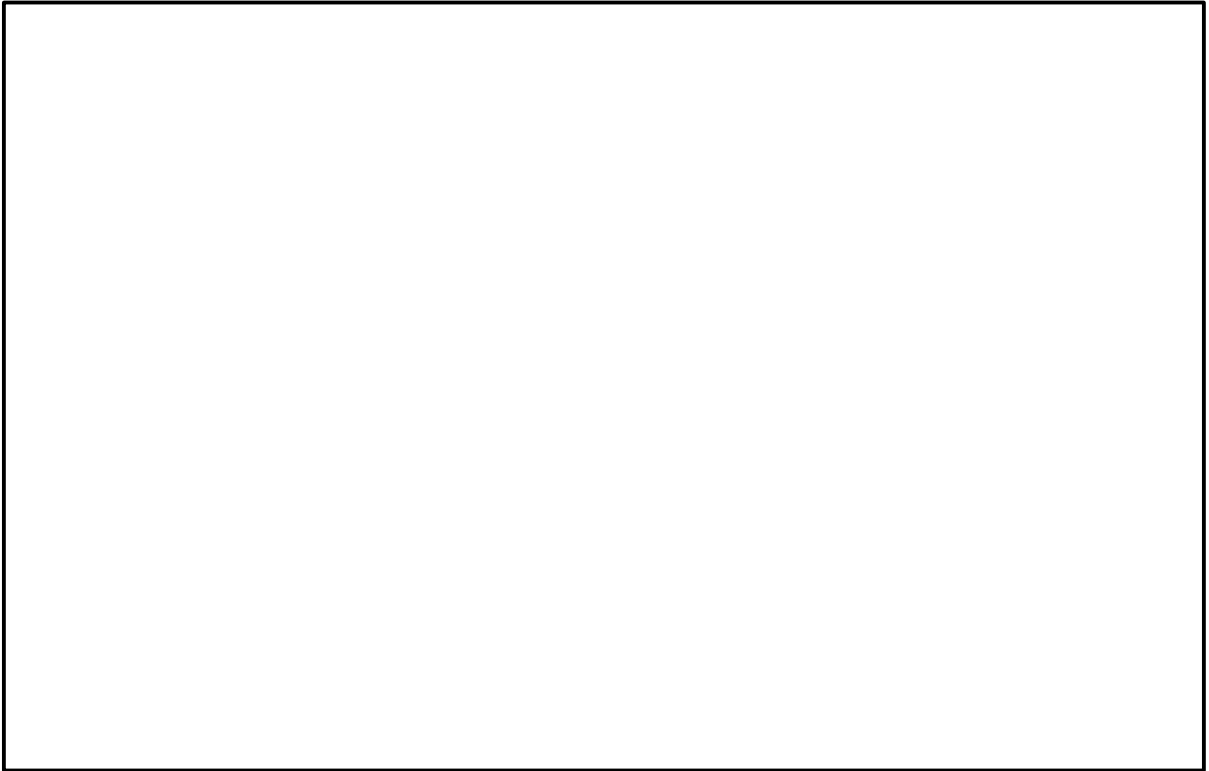
- What is the **exact** output for the program below?
- Print and use the tracing worksheet provided on the next page. You are required to trace the program by hand in order to get credit for the question. Show your work as well as the output on the tracing worksheet.
- Scan the tracing worksheet, then save it as **.pdf** file.
- Your output should go in the output box provided on the next page.

```
public class Tracing2
{
    public static void main(String[] args)
    {
        int k = 109;
        do
        {
            for(int i = 3; i < 9; i = i * 2)
            {
                if(k % i == 3)
                    k = k / 3;
                else
                    k = k / 2;
            }
            System.out.println(k);
        }while(k > 0);

        for(int i = 0; i < 2; i++)
        {
            for(int j = 0; j < 2; j++)
            {
                for(int m = 0; m < i * 2; m++)
                {
                    if(m == j && m == i)
                    {
                        System.out.println("i: " + i);
                        System.out.println("j: " + j);
                        System.out.println("m: " + m);
                    }
                }
            }
        }
    }
}
```

## Tracing Worksheet

Output:

A large, empty rectangular box with a black border, intended for the user to write or draw the output of their tracing exercise.

Memory box:

A large, empty rectangular box with a black border, intended for the user to write or draw information from their memory.

### **Problem 3:**

Create a class named Problem3, the program would do the following:

- The program should have a **main** method, for now just leave it empty.
- Create a method named **boxWithMinorDiagonal** which takes one integer **n** as parameter, and doesn't return any value, the method should print an **n-by-n** empty box of asterisks with the minor diagonal of '+' symbols. e.g, when **n = 9** the method would display:

```
*****
*           +*
*         + *
*       +  *
*     +   *
*   +    *
* +     *
*+      *
*****
```

- Create a method named **rightTriangle** which takes one integer **n** as parameter, and doesn't return any value, the method should print right triangle of asterisks. e.g, when **n = 9** the method would display:

```
*
**
* *
*  *
*   *
*    *
*     *
*      *
*       *
*****
```

- Create a method named **printNLetter** which takes one integer **n** as parameter, and doesn't return any value, the method should print an **n-by-n** letter N of asterisks. e.g, when **n = 9** the method would display:

```
*           *
**          *
*  *        *
*   *       *
*    *      *
*     *     *
*      *    *
*       *   *
*        **  *
*         *
```

- Create a method named **fancySquare** which takes one integer **n** as parameter, and doesn't return any value, the method should print an **n-by-n** shape shown below. e.g, when **n = 9** the method would display:

```

@*****@
*+      +*
*  +    +  *
*    +  +    *
*      @      *
*    +  +    *
*  +    +  *
*+      +*
@*****@

```

- Create a method named **plusInSquare** which takes one integer **n** as parameter, and doesn't return any value, the method should print an **n-by-n** shape shown below. e.g, when **n = 9** the method would display:

```

XXXXXXXXXX
X...+...X
X...+...X
X...+...X
X+++++++X
X...+...X
X...+...X
X...+...X
XXXXXXXXXX

```

- Go back to the main method and prompt the user to enter an integer **n**. The value of **n** must be an odd integer and greater than or equal 5.
- Call the methods created above and pass the integer **n** as an argument to those methods.
- Your methods should work for any integer **n**, so **DO NOT** hard code the print statements.
- Copy the output to a text file named Problem3.txt

## Problem 4:

- Create a class named Problem4, the program will prompt the user to enter a positive integer **n**, the program should find the sum of the digits of the number repeatedly until we get to a single digit.

### Examples:

If  $n = 7676$

$7 + 6 + 7 + 6 = 26$

Then  $2 + 6 = 8$

The program should display 8.

If  $n = 567432678$

$5 + 6 + 7 + 4 + 3 + 2 + 6 + 7 + 8 = 48$

Then  $4 + 8 = 12$

Then  $1 + 2 = 3$

The program should display 3.

- Below are two sample runs.
- Copy the output to a text file named Problem4.txt

```
Enter a positive integer: -1
Enter a positive integer: -3
Enter a positive integer: 6592
Single digit result: 4
```

```
Enter a positive integer: -6
Enter a positive integer: 63726195
Single digit result: 3
```

## General Instructions:

- No hard copies will be collected.
- Do not send your files through the email!
- You should submit your work by the due date, **No** extensions will be given. (See syllabus for late homework policy).
- **DO NOT** turn in multiple files, only one .zip file.

## What to turn in:

There should be three .java file, three .txt file and one .pdf file, put all those files into a zip file and name it <YourFirstName\_YourLastName>.zip, submit the zip file into the Dropbox on D2L.

How to zip multiple files?

On Windows: Select all the files > right click > Send to > Compressed File

On Mac: Select all the files > Click/Tap with two fingers > Compress Items