HW2 Code – Itay Haramati

```java
public class Divisors {
    public static void main (String[] args) {
        int x = Integer.parseInt(args[0]);
        for(int divisor = 1; divisor <= x; divisor++){
            if(x % divisor == 0){
                System.out.println(divisor);
            }
        }
    }
}
```

```java
/**
 * Prints a given string, backward. Then prints the middle character in the string.
 * The program expects to get one command-line argument: A string.
 */
public class Reverse {
    public static void main (String[] args){
        String str = args[0];
        String reverseString = "";
        int n = str.length();
        for(int i = n - 1; i >= 0; i--){
            reverseString = reverseString + str.charAt(i);
        }
        System.out.println(reverseString);
        System.out.println("The middle character is " + str.charAt((str.length() - 1) / 2));
    }
}
```

```java
/**
 *  Generates and prints random integers in the range [0,10),
 *  as long as they form a non-decreasing sequence.
 */
public class InOrder {
    public static void main (String[] args) {
        int num1;
        int num2 = (int)(10 * Math.random());
        int temp = num2;
        do{
            System.out.print(temp + " ");
            num1 = num2;
            num2 = (int)(10 * Math.random());
            temp = num2;
        }while(num2 >= num1);
    }
}
```

```java
/**
 *  Gets a command-line argument (int), and checks if the given number is perfect.
 */
public class Perfect {
    public static void main (String[] args) {
        int N = Integer.parseInt(args[0]);
        String s = N + " is a perfect number since " + N + " = " + 1;
        int sum = 1;
        for(int i = 2; i < N; i++){
            if(N % i == 0){
                s = s + " + " + i;
                sum += i;
            }
        }
        if (sum == N){
            System.out.println(s);
        }else{
            System.out.println(N + " is not a perfect number");
    }
}
}
```

```java
/**
 *  Gets a command-line argument n (int), and prints an n-by-n damka board.
 */
public class DamkaBoard {
    public static void main(String[] args) {
        int n = Integer.parseInt(args[0]);
        for(int i = 0; i < n; i++){
            for(int j = 0; j < n; j++){
                if(i % 2 == 0){
                    System.out.print("* ");}
                else{
                    System.out.print(" *");
                }
                }
            System.out.println("");}
            }
        }
```

```java
import java.util.Random;

/**
 *  Computes some statistics about families in which the parents decide
 *  to have children until they have at least one child of each gender.
 *  The program expects to get two command-line arguments: an int value
 *  that determines how many families to simulate, and an int value
 *  that serves as the seed of the random numbers generated by the program.
 *  Example usage: % java OneOfEachStats 1000 1
 */
public class OneOfEachStats {
        public static void main (String[] args) {
            // Gets the two command-line arguments
            int T = Integer.parseInt(args[0]);
            int seed = Integer.parseInt(args[1]);
            // Initailizes a random numbers generator with the given seed value
            Random generator = new Random(seed);


            //// In the previous version of this program, you used a statement like:
            //// double rnd = Math.random();
            //// Where "rnd" is the variable that stores the generated random value.
            //// In this version of the program, replace this statement with:
```

```
            //// double rnd = generator.nextDouble();
            //// This statement will generate a random value in the
range [0,1),
            //// just like you had in the previous version, except
that the
            //// randomization will be based on the given seed.
            //// This is the only change that you have to do in the
program.


            // User inputs T - number of families to form (amount of
runs)
            int countChildrenPerRun = 1;
            boolean isOneOfEach = false; // Tool to break while
            char firstChild = ' ';
            char child = firstChild;
            int totalCount = 0;
            int twoChilds = 0, threeChilds = 0, fourOrMoreChilds = 0;
// Counts the families
            // Each "for" creates one scenario
            for (int t = 0; t < T; t++) {
                // The "while" counts how many children until at least
                // one of each gender.
                firstChild = generator.nextDouble() > 0.5 ? 'b' : 'g';
                while (!isOneOfEach) {
                    child = generator.nextDouble() > 0.5 ? 'b' : 'g';
                    if (child != firstChild) {
                        isOneOfEach = true;
                    }
                    countChildrenPerRun += 1;
                }
                // Count families
```

```
                if (countChildrenPerRun == 2) twoChilds++;
                else if (countChildrenPerRun == 3) threeChilds++;
                else if (countChildrenPerRun > 3) fourOrMoreChilds++;
                totalCount += countChildrenPerRun;
                // Reset before next run
                isOneOfEach = false;
                countChildrenPerRun = 1;
            }
            double average = (double) (totalCount) / T;
            String mostCommon = "";
            if (twoChilds >= threeChilds && twoChilds >=
fourOrMoreChilds) {
                mostCommon = "2";
            } else if (threeChilds >= twoChilds && threeChilds >=
fourOrMoreChilds) {
                mostCommon = "3";
            } else {
                mostCommon = "4 or more";
            }
            System.out.println("Average: " + average +
                               " children to get at least one of each
gender.");
            System.out.println("Number of families with 2 children: "
+ twoChilds);
            System.out.println("Number of families with 3 children: "
+ threeChilds);
            System.out.println("Number of families with 4 or more
children: " +
                               fourOrMoreChilds);
            System.out.println("The most common number of children is
" +
                               mostCommon + ".");}}
```