Yarin Raichman

HW02

Divisors

```java
/**
 *  Gets a command-line argument (int), and prints all the divisors
of the given number.
 */
public class Divisors {
    // main(string[]) - the entry point of a Java program.
    // args - args contains the supplied command-line
    // arguments as an array of String objects.
    public static void main (String[] args) {
        // Gets a number form the user (converted from string to
integer).
        int num = Integer.parseInt(args[0]);
        // Checking what are teh divisors of the number that was
given by the user.
        for(int i = 1; i <= num; i++){
            // Checking if is a divisor of the number that was given
by the user.
            if ((num % i) == 0 ) {
                // Prining all of the divisors of the number that
was given by the user.
                System.out.println(i);
            }
        }
    }
}
```

Reversing a string

```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 {
    // main(string[]) - the entry point of a Java program.
    // args - args contains the supplied command-line
    // arguments as an array of String objects.
    public static void main (String[] args){
        // Gets the length of the string that was given by the user.
        int len = args[0].length();
        // Declaring an empty string that will contain the string
that we got from the user backwards.
        String revSting = "";
        // Inserting the backward string to the new string char by
char.
        for(int i = len - 1; i >= 0; i--){
            revSting += args[0].charAt(i);
        }
        // Printng the new reverse string.
        System.out.println(revSting);
        // Printing the middle character of the new string.
        System.out.println("The middle character is " +
revSting.charAt(len / 2));
    }
}
```

Lucky streak

```java
/**
 *  Generates and prints random integers in the range [0,10),
 *  as long as they form a non-decreasing sequence.
 */
public class InOrder {
    // main(string[]) - the entry point of a Java program.
    // args - args contains the supplied command-line
    // arguments as an array of String objects.
    public static void main (String[] args) {
        // Declaring 2 numbers that one is always amaller the the
other (to get inside the while)
        // and the other is a random generated number between 0 and
10.
        int gen1 = -1;
        int gen2 = (int)((Math.random() * 100) % 11);
        // Checking if the first generated number is smaller or
equal to the second one.
        while (gen1 <= gen2) {
            gen1 = gen2;
            // Generating a second number.
            gen2 = (int)((Math.random() * 100) % 11);
            // Printing teh generated number is the same line as the
others.
            System.out.print(gen1 + " ");
        }
    }
}
```

Perfect Numbers

```java
/**
 *  Gets a command-line argument (int), and chekcs if the given
number is perfect.
 */
public class Perfect {
    // main(string[]) - the entry point of a Java program.
    // args - args contains the supplied command-line
    // arguments as an array of String objects.
    public static void main (String[] args) {
        // Gets a number from the user (converted from string to
integer).
        int num = Integer.parseInt(args[0]);
        // Declaring a string to be all of the number's divisors
(every numebr is divisable by 1).
        String divisors = "1";
        // Declaring a variable to sum all of the number's divisors
(every numebr is divisable by 1).
        int sum = 1;
        // Calculating all of the numbers divisors undil it's half.
        for(int i = 2; i <= num/2; i++){
            // Checking if i is a divisor of the number we got from
the user.
            if (num % i == 0) {
                // Adding i to the string of divisors.
                divisors += " + " + i;
                // Adding i to the sum of the number's divisors.
                sum += i;
            }
        }
        // Checking if the nmber we got from the user is indeed a
perfect number.
        if (sum == num) {
            // Printing a message to show tha the number is perfect
with a proof as the question asks.
            System.out.println(num + " is a perfect number since " +
num + " = " + divisors);
        }
        else{
            // Printing a message to show tha the number is not
perfect as the question asks.
            System.out.println(num + " is not a perfect number");
        }
    }
}
```

Damka Board

```java
/**
 *  Gets a command-line argument n (int), and prints an n-by-n damka
board.
 */
public class DamkaBoard {
    // main(string[]) - the entry point of a Java program.
    // args - args contains the supplied command-line
    // arguments as an array of String objects.
    public static void main(String[] args) {
        // Gets a number to determine the size of the Damka board
from the user (converted from string to integer).
        int len = Integer.parseInt(args[0]);
        // Declaring and empty row;
        String row = "";
        // Adding * to the row as the number that was given by the
user.
        for (int i = len; i > 0; i--){
            if (i == 1) {
                row += "*";
            }
            else{
                row += "* ";
            }
        }
        // printing the rows in the way that was asked for in thr
question.
        for (int i = 0; i < len; i++){
            if (i % 2 == 0) {
                System.out.println(row + " ");
            }
            else{
                System.out.println(" " + row);
            }
        }
    }
}
```

One of Each Stats (final version)

```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
 */
        //// 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.
public class OneOfEachStats {
    // main(string[]) - the entry point of a Java program.
    // args - args contains the supplied command-line
    // arguments as an array of String objects.
    public static void main (String[] args) {
        // Gets the two command-line arguments (converted from
string to integer).
        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);
        // Declaring the variables that will be used to count each
category in the question.
        int twoChildren = 0;
        int threeChildren = 0;
        int fourOrMore = 0;
        // Declaring the variables that will be used to count the
over-all number of chldren and
```

```java
        // to calculate the most common group of the above 3.
        int sumChildren = 0;
        int maxChildren = 0;
        // Running the simulation as much times as the number we got
from the user.
        for(int i = 0; i < T; i++) {
            // Boolean variables for recording if there is at ;east
1 boy and 1 girl.
            Boolean atLeastOneGirl = false;
            Boolean atLeastOneBoy = false;
            // Declaring a gender variable which will be determined
by the random methood in a few lines.
            double gender = 0.0;
            // Declaring a variable that will be used to count the
number of children in this specific run.
            int children = 0;
            // Generating children until there is at least one of
each gender.
            while (!(atLeastOneBoy && atLeastOneGirl)) {
                // Generating a random number from 0 to 1 (using a
specific seed) to determine which gender is added.
                gender = generator.nextDouble();
                // Checking if the number that was generated means a
boy or a girl,
                // changing the boolean variables accordingly.
                if (gender > 0.5){
                    atLeastOneBoy = true;
                    children ++;
                }
                else{
                    atLeastOneGirl = true;
                    children ++;
                }
            }
            // countng each children group.
            if(children == 2){
                twoChildren ++;
            }
            if(children == 3){
                threeChildren ++;
            }
            if(children >= 4){
                fourOrMore ++;
            }
            // Adding the sum of the children in this run to the sum
of the over-all children.
            sumChildren += children;
        }
```

```
        // Printing the avarage children to get at least one of each
gender exactly as the question asks
        // (converted from integer to double).
        System.out.println("Average: " + ((double)sumChildren /
(double)T) + " children to get at least one of each gender.");
        // Printing the number of each children group exactly as the
question asks.
        System.out.println("Number of families with 2 children: " +
twoChildren);
        System.out.println("Number of families with 3 children: " +
threeChildren);
        System.out.println("Number of families with 4 or more
children: " + fourOrMore);
        // Calculating the most common children group.
        maxChildren = Math.max(twoChildren, Math.max(threeChildren,
fourOrMore));
        // Printing which group is the most common exactly as the
question asks.
        if (maxChildren == twoChildren) {
            System.out.println("The most common number of children
is 2.");
        }
        else if(maxChildren == threeChildren) {
            System.out.println("The most common number of children
is 3.");
        }
        else {
            System.out.println("The most common number of children
is 4 or more.");
        }
    }
}
```