Divisors

```java
/**
 *  Gets a command-line argument (int), and prints all the divisors of
the given number.
 */
public class Divisors {
    public static void main (String[] args) {
        // Parse the command-line argument as an integer
        int number = Integer.parseInt(args[0]);

        // Iterate through numbers from 1 to the given number and
prints the divisors
        for (int i = 1; i <= number; i++) {
            if (number % i == 0) {
                System.out.println(i);
            }
        }
    }
}
```

Reverse

```
/**
 * 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) throws Exception {

        // Get the input string from the command-line argument and
declare some args
        String word = args[0];
        String reverseWord = "";
        int middleIndex = 0;

        // Calculate the middle index of the string
        middleIndex = (word.length()%2 == 0) ? (word.length()/2) :
(word.length()/2) + 1;

        // Reverse the string
        for (int i = word.length() - 1; i >= 0; i--) {
            reverseWord = reverseWord + word.charAt(i);
        }

        // Print the reversed string and the middle char of the string
        System.out.println(reverseWord);
        System.out.println("The middle character is " +
word.charAt(middleIndex - 1));
    }
}
```

InOrder

```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) {
        // Generate the two integers
        int num1 = (int) (Math.random()*(9)) + 1;
        int num2 = (int) (Math.random()*(9)) + 1;

        // Print the first random integer
        System.out.print(num1 + " ");

        // Generate and print subsequent random integers in a non-
decreasing sequence
        while (num2>=num1) {
            System.out.print(num2 + " ");
            num1 = num2;
            num2 = (int) (Math.random()*(9)) + 1;
        }
    }
}
```

DamkaBoard

```
/**
 *  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 num = Integer.parseInt(args[0]);
    for (int i = 0; i < num; i++) {
        if (i % 2 ==0) {
            for (int j = 0; j < num; j++) {
                System.out.print("* ");
            }
        }
        else {
            for (int j = 0; j < num; j++) {
                System.out.print(" *");
            }
    }
    System.out.println();
    }
}
}
```

Perfect

```java
/**
 *  Gets a command-line argument (int), and chekcs if the given number is perfect.
 */
public class Perfect {
    public static void main(String[] args) throws Exception {

        // Parse the command-line argument as an integer and declare some args
        int num = Integer.parseInt(args[0]);
        int sum = 1;
        String strPerfect = num + “ is a perfect number since “ + num + “ = 1”;

        // Iterate through potential divisors up to (num – 1) and check if I is a divisor of num
        for (int i=2; i<num; i++) {
            if (num % I == 0) {
                sum += I;
                strPerfect = strPerfect + " + " + i;
            }
        }

        // Check if the sum of divisors equals the original number
        if (sum == num) {
            System.out.print(strPerfect);
        }
        else {
            System.out.println(num + " is not a perfect number");
        }
    }
}
```

OneOfEachStats

```
import java.util.Random;

/**
 * Computes statistics about families in which the parents decide
 * to have children until they have at least one child of each gender.
 * The program expects two command-line arguments: the number of
families to simulate
 * and the seed for the random number generator.
 * Example usage: % java OneOfEachStats 1000 1
 */
public class OneOfEachStats {
    public static void main(String[] args) {
        // Parse command-line arguments
        int numFamiliesToSimulate = Integer.parseInt(args[0]);
        int randomSeed = Integer.parseInt(args[1]);

        // Initialize variables
        int totalExperiments = 0;
        int totalChildren = 0;
        int childrenInCurrentFamily = 0;
        int familiesWithTwoChildren = 0;
        int familiesWithThreeChildren = 0;
        int familiesWithFourOrMoreChildren = 0;
        boolean hasGirl = false;
        boolean hasBoy = false;

        // Initialize a random number generator with the given seed
value
        Random randomGenerator = new Random(randomSeed);

        // Simulate different cases of families until they have a boy
and a girl
        while (totalExperiments < numFamiliesToSimulate) {
            while (!(hasGirl && hasBoy)) { // Check until there is a
boy and a girl in the family
                double randomGender = randomGenerator.nextDouble();
                if (randomGender < 0.5) {
                    hasGirl = true;
                } else {
                    hasBoy = true;
                }
                totalChildren++;
                childrenInCurrentFamily++;
            }
```

```
            // Update statistics based on the number of children in
the family
            if (childrenInCurrentFamily == 2) {
                familiesWithTwoChildren++;
            } else if (childrenInCurrentFamily == 3) {
                familiesWithThreeChildren++;
            } else {
                familiesWithFourOrMoreChildren++;
            }

            // Reset flags and counters for the next family
            totalExperiments++;
            hasGirl = false;
            hasBoy = false;
            childrenInCurrentFamily = 0;
        }

        // Calculate the average number of children in a family
        double averageChildren = (double) totalChildren / (double)
numFamiliesToSimulate;

        // Output the results
        System.out.println("Average: " + averageChildren + " children
to get at least one of each gender.");
        System.out.println("Number of families with 2 children: " +
familiesWithTwoChildren);
        System.out.println("Number of families with 3 children: " +
familiesWithThreeChildren);
        System.out.println("Number of families with 4 or more
children: " + familiesWithFourOrMoreChildren);

        // Determine the most common number of children in a family
        if (familiesWithTwoChildren >= familiesWithThreeChildren &&
familiesWithTwoChildren >= familiesWithFourOrMoreChildren) {
            System.out.println("The most common number of children is
2.");
        } else if (familiesWithThreeChildren > familiesWithTwoChildren
&& familiesWithThreeChildren >= familiesWithFourOrMoreChildren) {
            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.");
        }
    }
```

```
}
```