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) {
        // takes int from command line and initializes divisor
        int x = Integer.parseInt(args[0]);
        int d = 0;
        // loop tests for divisors and prints if valid
        while (d < x) {
            if (x % ++d == 0) {
                System.out.println(d);
            }
        }
    }
}
```

REVERSE

```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){
        // take input for string and determines middle char
        String s = args[0];
        int i = s.length() - 1;
        char mc = s.charAt(i / 2);
        // loops through string from last position, prints each char, then prints
middle char
        while (i >= 0) {
            System.out.print(s.charAt(i));
            i--;
        }
        // prints middle character of string
        System.out.println("\nThe middle character is " + mc);
    }
}
```

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) {

        // initialize a random int and an int less than zero
        int n = (int) (Math.random() * 10);
        int m = -1;
        // prints n while larger than m, sets m to n, then sets n to new random
int
        while (n >= m) {
            System.out.print(n);
            m = n;
            n = (int) (Math.random() * 10);
            if (n >= m) { // tests whether space is needed to delimit output
value from next output value
                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) {
        // takes int from command, initializes sum, constructs start of string
for affirmative result
        int n = Integer.parseInt(args[0]);
        int sum = 1;
        String p = n + " is a perfect number since " + n + " = 1";
        // loops for divisors, adding them to sum if valid
        for (int d = 2; d < n; ++d) {
            if (n % d == 0 ) {
                p += " + " + d;
                sum += d;
            }
        }
        // prints affirmative message if n is perfect, else negative message
        if (sum == n && n > 1) {
            System.out.println(p);
        } else {
            System.out.println(n + " is not a perfect number");
        }
    }
}
```

DAMKABOARD

```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) {
        // takes int and initializes iterators
        int n = Integer.parseInt(args[0]);
        int i = 0;
        int j = 0;
        boolean opposite = true; // used to change pattern each new row
        // loops through rows and columns to print checkerboard pattern
        while (i++ < n) {
            while (j++ < n) {
                if (opposite) {
                    System.out.print("* ");
                } else {
                    System.out.print(" *");
                }
            }
            System.out.println();
            j = 0;
            opposite = !opposite; // changes pattern
        }
    }
}
```

ONEOFEACH

```
/**
 *  Simulates the formation of a family in which the parents decide
 *  to have children until they have at least one child of each gender.
 */
public class OneOfEach {
    public static void main (String[] args) {
        // initialize counts for girls and boys born
        int g = 0;
        int b = 0;
        // loops while either value is still 0 and prints result of birth
        while (g == 0 || b == 0) {
            if (g > 0 || b > 0) {
                System.out.print(' ');
            }
            if ((int) (Math.random() * 2) == 1) {
                System.out.print('b');
                b++;
            } else {
                System.out.print('g');
                g++;
            }
        }
        // prints total children
        System.out.println("\nYou made it... and you now have " + (g + b) + "
children.");
    }
}
```

ONEOFEACHSTATS1

```
/**
 *  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 one command-line argument: an int value
 *  that determines how many families to simulate.
 */
public class OneOfEachStats1 {
    public static void main (String[] args) {
        // Gets the two command-line arguments
        int T = Integer.parseInt(args[0]);
        // initailizes gender counts and iterator
        int g = 0;
        int b = 0;
        int i = 0;
        int count = 0; // number of children per test
        int total = 0;
        // numbers of tests resulting in corresponding values
        int two = 0;
        int three = 0;
        int fourPlus = 0;
        String mode = "2";
        // loops T number of times running each experiment until at least one of
each gender born
        while (i < T){
            while (g == 0 || b == 0) {
                if (g > 0 || b > 0) {
                }
                if ((int) (Math.random() * 2) == 1) {
                    b++;
                } else {
                    g++;
                }
                count++;
            }
            if (count == 2) {
                two++;
            } else if (count == 3) {
                three++;
            } else {
                fourPlus++;
            }

            total += count;
```

```
            count = 0;
            g = 0;
            b = 0;
            i++;
        }
        // determines most common number of children for a test
        if (three > two && three >= fourPlus) {
            mode = "3";
        } else  if (fourPlus > two && fourPlus > three) {
            mode = "4 or more";
        }
        // prints results of data
        System.out.println("Average: " + ((double) total / T) + " children to get
at least one of each gender.");
        System.out.println("Number of families with 2 children: " + two);
        System.out.println("Number of families with 3 children: " + three);
        System.out.println("Number of families with 4 or more children: " +
fourPlus);
        System.out.println("The most common number of children is " + mode);
    }
}
```

ONEOFEACHSTATS

```
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);
        // initailizes gender counts and iterator
        int g = 0;
        int b = 0;
        int i = 0;
        int count = 0; // number of children per test
        int total = 0;
        // numbers of tests resulting in corresponding values
        int two = 0;
        int three = 0;
        int fourPlus = 0;
        String mode = "2";
        // loops T number of times running each experiment until at least one of
each gender born
        while (i < T){
            while (g == 0 || b == 0) {
                if (g > 0 || b > 0) {
                }
                if ((int) (generator.nextDouble() * 2) == 1) {
                    b++;
                } else {
                    g++;
                }
                count++;
            }
            if (count == 2) {
                two++;
            } else if (count == 3) {
```

```
                three++;
            } else {
                fourPlus++;
            }

            total += count;
            count = 0;
            g = 0;
            b = 0;
            i++;
        }
        // determines most common number of children for a test
        if (three > two && three >= fourPlus) {
            mode = "3";
        } else  if (fourPlus > two && fourPlus > three) {
            mode = "4 or more";
        }
        // prints results of data
        System.out.println("Average: " + ((double) total / T) + " children to get
at least one of each gender.");
        System.out.println("Number of families with 2 children: " + two);
        System.out.println("Number of families with 3 children: " + three);
        System.out.println("Number of families with 4 or more children: " +
fourPlus);
        System.out.println("The most common number of children is " + mode +
".");

        //// 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.

    }
}
```