```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) {
        // gets an integer from the user
        int x = Integer.parseInt(args[0]);
        // checks if x is positive or negative

        if ( x >= 0 ) {
            for (int i = 1; i <= x; i++) { // checks if i is a divisor of x then prints i
                 if (x % i == 0) {
                    System.out.println(i);
                }
            }
        } else {
            for (int i = -1; i >= x; i--) {
                if (x % i == 0) {
                    System.out.println(i * (-1)); //checks if i is a divisor of x then prints i as positive value
                }
            }
        }

    }
}

```

```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 originalStr = args[0];
        String reverseStr = "";
        char mid = '0';
        int n = originalStr.length();

        for (int i = n - 1 ; i >= 0 ; i--) {
            // reverse the orginal string
            reverseStr += originalStr.charAt(i) ;
            // find the middle character in the original string
            if ( i == (n - 1) / 2) {
                mid = originalStr.charAt(i);
            }
        }

        // prints the reversed string
        System.out.println(reverseStr);
        // prints the mid character
        System.out.println("The middle character is " + mid);

    }
}

```

```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 last;
        // first random int
        int num = (int) (Math.random() * 10);
        do {
            // keeps the last random int
            last = num;
            System.out.print(num + " ");
            // randomizing new int
            num = (int) (Math.random() * 10) ;
        } while (num >= last); // checks if there is a non-decreasing sequence
    }
}

```

```java
/**
 *  Gets a command-line argument (int), and chekcs if the given number is perfect.
 */
public class Perfect {
    public static void main (String[] args) {
        int n = Integer.parseInt(args[0]);
        int sum = 1;
        String str = n + " is a perfect number since " + n + " = " + 1;
        // checks for all of n's divisors
        for (int i = 2; i < n; i++) {
            if (n % i == 0) {
                sum += i; // sums all the divisors
                str += " + " + i; // adds every divisor to the answer
            }
        }
        // checks if n is perfect and prints the answer
        if (sum == n) {
            System.out.println(str);
        } 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) {
        // gets board size from the user
        int n = Integer.parseInt(args[0]);
        // prints the board
        for (int i = 1; i <= n ; i++ ) {
            // prints even lines
            if ( i % 2 == 0) {
                for (int j = 0; j < n ; j++ ) {
                    System.out.print(" *");
                }
            } else {     // prints odd lines
                for (int j = 0; j < n ; j++ ) {
                    System.out.print("* ");
                }
            }
            // moves to a new line
            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);
        double sum = 0;
        // counters for number of families with x children
        int twoChildCount = 0;
        int threeChildCount = 0;
        int fourPlusChildCount = 0;
        String mode = "2.";
        // loop of T experiments
        for (int i = 0; i < T ; i++) {
            boolean isBoy = false;
            boolean isGirl = false;
            int count = 0;
            // loop running until there are 1 boy and 1 girl
            while (!isBoy || !isGirl) {
                double rnd = generator.nextDouble();   // randomizing a child
                if (rnd > 0.5) {    // checks if child is a girl
                    isGirl = true;
                } else {
                     isBoy = true;
                }
                // counts how many children were born
                count++;
            }

            // count how many families were born with the same number of
            children
            if (count >= 4) {
                fourPlusChildCount++;
            } else if (count == 3) {
                threeChildCount++;
            } else {
                twoChildCount++;
            }
            // sum of all children that were born in T experiments
            sum += count;
        }


```

```
54          // checks mode
55          if (threeChildCount > twoChildCount && threeChildCount >
            fourPlusChildCount) {
56              mode = "3.";
57          } else if (fourPlusChildCount > threeChildCount && fourPlusChildCount >
            twoChildCount) {
58              mode = "4 or more.";
59          }
60          // prnts the results
61          System.out.println("Average: " + (sum / T) +
62          " children to get at least one of each gender.");
63          System.out.println("Number of families with 2 children: " +
            twoChildCount);
64          System.out.println("Number of families with 3 children: " +
            threeChildCount);
65          System.out.println("Number of families with 4 or more children: " +
            fourPlusChildCount);
66          System.out.println("The most common number of children is " + mode);
67          //// In the previous version of this program, you used a statement like:
68          //// double rnd = Math.random();
69          //// Where "rnd" is the variable that stores the generated random value.
70          //// In this version of the program, replace this statement with:
71          //// double rnd = generator.nextDouble();
72          //// This statement will generate a random value in the range [0,1),
73          //// just like you had in the previous version, except that the
74          //// randomization will be based on the given seed.
75          //// This is the only change that you have to do in the program.
76
77      }
78  }
79
```