```
public class Divisors {

    public static void main(String[] args) {

        // init the param of x
        int x = Integer.parseInt(args[0]);

        // init a loop that runs x times from x to 1
        for (int d = 1; d <= x ; d++) {
            // m has the remainder of dividing x / d
            int m = x % d;

            // check if it has no remainder - a divisor
            if (m == 0){
                // print the divisor
                System.out.println(d);
            }

        }


    }

}
```

```
public class Reverse {

    public static void main(String[] args) {

        // init the original string that was given from user
        String original = args[0];

        // init the reversed sting
        String reversed = "";

        // get the length os the string using the function
        int length = original.length();

        // get the mid index of the string
        int mid = (length - 1) / 2;

        // init the mid char of the string
        char mid_char = '!';

        // init a loop that runs from the end to the start of the string
        for (int i = length - 1; i >=0 ; i--) {

            // get the current letter using the function
            char current_letter = original.charAt(i);

            // check if it is now the current letter
            if (i == mid){

                // set the value of the current letter
                mid_char = current_letter;
            }

            // add the current letter to the current result
            reversed = reversed + current_letter;
        }


        // print the results
        System.out.println(reversed);
        System.out.println("The middle character is " + mid_char);
    }

}
```

```
public class InOrder {
    public static void main(String[] args) {

        // init a random number between 0 - 10
        int current_number = (int) (Math.random() * 11);

        // init the value of the following number
        // the value is not between the range in order to check if it is the first time
        int next_number = -1;

        // init do while loop that runs while the next_number > current_number
        do {
            // check if it is the first iteration
            if (next_number != -1 ){
                // if not the first time - the current number is the next one form the previous iteration
                current_number = next_number;
            }
            // print the current number
            System.out.println(current_number);
            // set a random value of 0 - 10 to the following number at the series
            next_number = (int) (Math.random() * 11);
        }
        // check if the next one is greater
        while (next_number > current_number);

    }
}
```

```
public class DamkaBoard {
    public static void main(String[] args) {
        // init n as the requested number of rows and columns
        int n  = Integer.parseInt(args[0]);

        // init the variable that check if i need to add space at the beginning or at the end of a line
        boolean space_at_start = false;

        // init a loop that runs n times (every line)
        for (int i = 0; i < n; i++) {

            // check if need to add space at the start
            if (space_at_start == true){
                // add space
                System.out.print(" ");
            }
            // init a loop that runs n time that prints a line
            for (int j = 0; j < n; j++) {

                // dont unnecessary space at the end
                if (j == n - 1 && space_at_start){
                    System.out.print("*");
                } else {
                    System.out.print("* ");
                }
            }
            // change the boolean to the opposite value
            space_at_start = !space_at_start;
            // get one line down

            System.out.println();
        }

        System.out.println();

    }
}
```

```
public class Perfect {
    public static void main(String[] args) {

        // init the parameter that holds the given value from the user
        int parameter = Integer.parseInt(args[0]);

        // init the start of success message
        String equation = parameter + " is a perfect number since " + parameter + " = 1";

        // init the sum of dividers - 1 always divides
        int sum = 1;

        // init a loop that runs from 2 to param -1
        for (int i = 2; i < parameter; i++) {

            // check if the current number divides the parameter with no remainder
            if ( parameter % i == 0){

                // add i to the sum
                sum += i;

                // add i to the equation
                equation = equation + (" + " + i);
            }
        }

        // check if perfect number
        if(sum == parameter){

            // print the final equation
            System.out.println(equation);
        }else{

            // print the fail message
            String message_not_perfect = parameter +" is not a perfect number";
            System.out.println(message_not_perfect);
        }

    }
}
```

```
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);

      // define t as parameter from args
      int t = Integer.parseInt(args[0]);
      // init the value of sum of born children (to calculate the average)
      int sum_children = 0;
      // init the variable that counts number of families with 2 children
      int two_children = 0;
      // init the variable that counts number of families with 3 children
      int three_children = 0;
      // init the variable that counts number of families with 4 or more children
      int four_or_more = 0;
      for (int i = 0; i < t; i++) {
         // init the boolean variable that indicates if a boy was born with false (a boy wasnt born yet)
         boolean have_boy = false;
         // init the boolean variable that indicates if a girl was born with false (a girl wasnt born yet)
         boolean have_girl = false;
         // init the variable the count of children in each family
         int count = 0;
         // do the commands inside the while if a family have at least one boy and girl yet
         while (have_boy == false || have_girl == false){
            // init variable that generates random number - 0 or 1
            int x = (int) (generator.nextDouble() * 2);

            // 0 if a girl was born
            if (x == 0){
               // check if a girl was born yet
               if (have_girl == false){
                  // change the value because a girl was just born
                  have_girl = true;
               }
               // if a boy was born
            } else {
               // check if a boy was born yet
               if (have_boy == false){
                  // change the value because a boy was just born
```

```
            have_boy = true;
          }
        }
        // increase the count of born children
        count ++;
      }
      // add the sum of the count of children of this current family
      sum_children += count;
      // check the value of count
      if (count == 2){
        // increase the value of number of families with 2 children
        two_children ++;
      } else {
        if (count == 3){
          // increase the value of number of families with 3 children
          three_children ++;
        } else {
          // increase the value of number of families with 4 or more children
          four_or_more ++;
        }
      }

    }
    // calculating the value average children per family and casting the result to double
    double avg = (double)(sum_children)/t;
    // printing the results
    System.out.println("Average: " + avg + " children to get at least one of each gender.");
    System.out.println("Number of families with 2 children: " + two_children);
    System.out.println("Number of families with 3 children: " + three_children);
    System.out.println("Number of families with 4 or more children: " + four_or_more);

    // get the maximum value of the 3 variables using math.max function
    int most_common = Math.max(Math.max(two_children, three_children), four_or_more);

    // check if the most common case is family with 2 children
    if (most_common == two_children){
      // print result
      System.out.println("The most common number of children is 2.");
    } else {
      // check if the most common case is family with 3 children
      if (most_common == three_children){
        // print result
        System.out.println("The most common number of children is 3.");
      } else {
        // check if the most common case is family with 4 or more children - default case
        // print result
        System.out.println("The most common number of children is 4 or more.");
      }
    }

    System.out.println();
```

```
    }
}
```