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

        int number = Integer.parseInt(args[0]);

        for (int i = 1; i <= number; i++) {
            if (number % i == 0) {
                System.out.println(i);
            }
        }
    }
}
```

```
public class Reverse {
    public static void main (String[] args){
        String word = args[0];
        String reversedWord = "";

        for (int i = word.length() - 1; i >= 0; i--) {
            reversedWord += word.charAt(i);
        }

        int middleIndex = reversedWord.length() / 2;
        char middleChar = reversedWord.charAt(middleIndex);

        System.out.println(reversedWord);
        System.out.println("The middle character is " +
middleChar);
    }
}
```

```
public class InOrder {
    public static void main (String[] args) {
        int number = 0;
        int nextNumber;

        do {
            nextNumber = (int)(Math.random() * 10);
            if (nextNumber >= number) {
                System.out.print(nextNumber + " ");
                number = nextNumber;
            } else {
                break;
            }
        } while (true);
  }
}
```

```
public class DamkaBoard {
    public static void main(String[] args) {
        int n = Integer.parseInt(args[0]);

        for (int i = 0; i < n; i++) {
            for (int j = 0; j < n; j++) {
                if (i % 2 == 0) {
                    System.out.print("* ");
                } else {
                    System.out.print(" *");
                }
            }
            System.out.println();
            }
        }
    }
```

```
public class Perfect {
    public static void main (String[] args) {
        int N = Integer.parseInt(args[0]);
        int sumOfDivisors = 1; // Include 1 as a divisor
        String divisors = "";

        for (int i = 2; i < N; i++) {  // Start from 2 to avoid
duplicates
            if (N % i == 0) {
                sumOfDivisors += i;
                divisors += " + " + i;
            }
        }

        if (sumOfDivisors == N) {
            System.out.println(N + " is a perfect number since "
+ N + " = 1" + divisors);
        } else {
            System.out.println(N + " is not a perfect number");
        }
    }
}
```

```
import java.util.Random;
public class OneOfEachStats {
    public static void main (String[] args) {

        int T = Integer.parseInt(args[0]);
        int seed = Integer.parseInt(args[1]);
        Random generator = new Random(seed);

        int totalChildren = 0; // Stores the total number of
children across all simulated families.
        int familiesWith2 = 0; // Counts families with 2
children.
        int familiesWith3 = 0; // Counts families with 3
children.
        int familiesWith4OrMore = 0; // Counts families with 4 or
more children.
        int mostCommonCount = 0; // Stores the highest count of
families with a particular number of children.
        int mostCommonNumberOfChildren = 0; // Stores the most
common number of children.

        for (int i = 1; i <= T; i++) { // Repeats the simulation
for T families.
            int sumOfChildren = 0;
            int boys = 0;
            int girls = 0;
            do { // Continues generating children until there's
at least one boy and one girl.
                if (generator.nextDouble() <= 0.5) {
                    boys++;
                } else {
                    girls++;
                }
                sumOfChildren = boys + girls;
            } while (boys < 1 || girls < 1);
            totalChildren += sumOfChildren;
            if (sumOfChildren == 2) {
                familiesWith2++;
                if (familiesWith2 > mostCommonCount) {
                    mostCommonCount = familiesWith2;
                    mostCommonNumberOfChildren = 2;
                }
            } else if (sumOfChildren == 3) {
                familiesWith3++;
                if (familiesWith3 > mostCommonCount) {
                    mostCommonCount = familiesWith3;
```

```
                    mostCommonNumberOfChildren = 3;
                }
            } else {
                familiesWith4OrMore++;
                if (familiesWith4OrMore > mostCommonCount) {
                    mostCommonCount = familiesWith4OrMore;
                    mostCommonNumberOfChildren = 4; // Or more,
as needed.
                }
            }
        }

        double average = (double) totalChildren / T; //
Calculates the average number of children per family.
        System.out.println("Average: " + average + " children to
get at least one of each gender.");
        System.out.println("Number of families with 2 children: "
+ familiesWith2);
        System.out.println("Number of families with 3 children: "
+ familiesWith3);
        System.out.println("Number of families with 4 or more
children: " + familiesWith4OrMore);
        System.out.println("The most common number of children is
" + mostCommonNumberOfChildren + ".");
    }
}
```