```
/**
 *  Gets a command-line argument (int), and prints all the divisors of the given number.
 */
public class Divisors {
	public static void main (String[] args) {
		int x= Integer.parseInt(args[0]);
		for (int i = 1; i <= x; i++)
		{
			if (x % i == 0) System.out.println(i);
		}
	}
}
```

```
/**
 * 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 a = args[0];
	int length = a.length();
	String bwd = "";
	char middle = 0;
	for (int i = 0; i < (length); i++)
	{
	if (i == (length / 2)) {
		middle = (length % 2 ==0) ? a.charAt(i-1) : a.charAt(i);
	}
	bwd= a.charAt(i) + bwd;
	}
	System.out.println(bwd);
	System.out.println("The middle character is " + middle);
	}
}
```

```
/**
 *  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 x= 0;
        int y= -1;
        do {
          x= (int)(10 * (Math.random()));
          if (x >= y) {
            System.out.print(x + " ");
            y= x;
           }
   }
        while (x >= y);
        }
}
```

```
/**
 *  Gets a command-line argument (int), and checks if the given number is perfect.
 */
public class Perfect {
	public static void main (String[] args) {
	int x= Integer.parseInt(args[0]);
	String perfect= x + " is a perfect number since " + x + " = 1";
	int div= 1;
	for (int i = 2; i < x; i++)
	// A loop that puts the divisors of the argument into string 'perfect' and addition	them to int 'div'
		{
			if (x % i == 0) {
			  perfect= perfect + " + " + i;
			  div= div + i;
		    }
	}
	if (div == x) {
	   System.out.println(perfect);
	}
	else {
	   System.out.println (x + " is not a perfect number");
	}
}
}
```

```
/**
 *  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 x= Integer.parseInt(args[0]);
	for (int i= 0; i < x; i= i + 2) { // A loop that prints 2 rows of x *
	   System.out.println ();
	   for (int t= 0; t < x; t++) { // A loop that prints a row of x *
	      System.out.print ("* ");
	   }
	   System.out.println ();
	   for (int p= 0; p < x; p++) { // A loop that prints an opposite row of x *
	      System.out.print (" *");
	   }
	}
}
}
```

```
/**
 *  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) {
        String children= "";
        boolean gender= true;
        boolean secGender= true;
        int amount= 1;
        if (Math.random() < 0.5) { // A loop that gets a randon gender
          children= "g";
          }
        else { gender= false;
            secGender= false;
      children= "b";
    }
        while (gender == secGender) { // A loop that gets randon genders until it gets a different one
                amount++;
            if (Math.random() < 0.5) {
              children= children + " g";
              secGender= true;
            }
            else {
                children=children + " b";
                secGender=false;
            }
            }
            System.out.println(children );
            System.out.println("You made it... and you have " + amount + " children.");

        }
  }
```

```
/**
 *  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) {
                int t= Integer.parseInt(args[0]);
                double avg= 0.0;
                int two= 0;
                int three= 0;
                int four= 0;

                for (int i= 0; i < t; i++) {
        boolean gender= true; //OneOfEach program starts (without printing)
        boolean secGender= true;
        int amount= 1;
        if (Math.random() > 0.5) { // A loop that get a randon gender
           gender= false;
           secGender= false;
     }
        while (gender == secGender) { // A loop that gets randon genders until it gets a different one
                amount++;
           if (Math.random() < 0.5) {
             secGender= true;
             }
     else {
                secGender= false;
       }
           }
            if (amount == 2) two++;
            if (amount == 3) three++;
            if (amount >= 4) four++;
```

```
                avg= avg + ((double)(amount));
            }
            avg= (avg / ((double)(t)));
            int common= 0;
            if (two >= three && two >= four) {
       common= 2;
       }
     if (three >= four && three >= two) {
                        common= 3;
       }
    if (four >= three && four >= two) {
            common= 4;
    }
            System.out.println("Average: " + avg + " 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: " + four);
            System.out.println("The most common number of children is " + common + ".");
            }
}
```

```
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
 */
/**
 *  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 OneOfEachStats {
  public static void main (String[] args) {
                    int t= Integer.parseInt(args[0]);
                    int seed = Integer.parseInt(args[1]);
                    Random generator = new Random(seed);
                    double avg= 0.0;
                    int two= 0;
                    int three= 0;
                    int four= 0;

                    for (int i= 0; i < t; i++) {
        boolean gender= true; /**OneOfEach program starts (without printing)**/
        boolean secGender= true;
        int amount= 1;
  double rnd= generator.nextDouble();
        if (rnd > 0.5) { // A loop that get a randon gender
           gender= false;
           secGender= false;
     }
```

```
            while (gender == secGender) { // A loop that gets randon genders until it gets a different one
                    rnd= generator.nextDouble();
                    amount++;
               if (rnd < 0.5) {
                 secGender= true;
                 }
      else {
                    secGender= false;
        }
              }
                 if (amount == 2) two++;
                 if (amount == 3) three++;
                 if (amount >= 4) four++;
                 avg= avg + ((double)(amount));
            }
            avg= (avg / ((double)(t)));
            int common= 0;
            if (two >= three && two >= four) {
      common= 2;
      }
   if (three >= four && three >= two) {
                    common= 3;
      }
  if (four >= three && four >= two) {
            common= 4;
  }
            System.out.println("Average: " + avg + " 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: " + four);
            System.out.println("The most common number of children is " + common + ".");
            }
}
```