```
/**
 *  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 x = args[0];
		int z = x.length();
		for (int i = 0; i < z; i++) {
			System.out.print(x.charAt((z - i) - 1));
		}
		System.out.println("");
		if ((z%2) == 0) {
			z--;
		}
		System.out.println("The middle character is " + (x.charAt(z / 2)));
	}
}
```

```
/**
 *  Generates and prints random integers in the range [0,10),
 *  as long as they form a non-decreasing sequence.
 */

import java.util.*;

public class InOrder {
        public static void main (String[] args) {
                int Min = 0;
                int Max = 10;
                int z;
                int y;
                Boolean n = true;
                int x = Min + (int)(Math.random() * (Max - Min));
                y = x;
                System.out.print(y);
                while (n==true) {
                z = Min + (int)(Math.random() * (Max - Min));
                        if (z >= x) {
                                System.out.print(" " + z);
                                x = z;
                        } else {
                                n = false;
                        }
                }
        }
}
```

```
/**
 *  Gets a command-line argument (int), and chekcs if the given number is perfect.
 */


public class Perfect {
	public static void main (String[] args) {
		int x = Integer.parseInt(args[0]);
		int y = 0;
		String Print = "";
		for (int i = 1; i <= (x-1); i++) {
			if ((x % i) == 0) {
				y = y + i;
				if (i != 1) {
					Print = Print + " + " + i;
				} else {
					Print = "" + i;
				}
			}
		}
		Boolean n = true;
		n = (y == x) ? true : false;
		if (n==true) {
			System.out.println(x + " is a perfect number since " + x + " = " +
Print);
		} 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]);
                String z = "*";
                for (int i = 1; i < x; i++) {
                        z = z + " *";
                }
                for (int j = 0; j < x; j++) {
                        if ((j%2) == 0) {
                                System.out.println(z + " ");
                        } else {
                                System.out.println(" " + z);
                        }
                }
        }
}
```

```
/**
 *  Simulates the formation of a family in which the parents decide
 *  to have children until they have at least one child of each gender.
 */
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]);
		double Z = T;
		Random generator = new Random(seed);
		double sum = 0;
		int TwoChildren = 0;
		int ThreeChildren = 0;
		int FourChildren = 0;
		int Min = 0;
		int Max = 1;
		double x = 0;
		double y = 0;
		for (int i = 0; i < T; i++) {
			x = generator.nextDouble();
			y = 0;
			Boolean boy = false;
			Boolean girl = false;
			int boyNum = 0;
			int girlNum = 0;
			int famSize = 0;
			String fam = "";
			if (x < 0.5) {
				boy = true;
				boyNum++;
				fam = "b ";
			} else {
				girl = true;
				girlNum++;
				fam = "g ";
			}
			Boolean n = false;
			while (n == false) {
				y = generator.nextDouble();
				if (y < 0.5) {
					if ((boy == true) && (girl == false)) {
						boyNum++;
						fam = fam + "b ";
					} else if ((boy == false) && (girl == true)) {
						boyNum++;
						fam = fam + "b ";
						n = true;
					}
				}
				if (y >=0.5) {
					if ((girl == true) && (boy == false)) {
						girlNum++;
						fam = fam + "g ";
					} else if ((girl == false) && (boy == true)) {
						girlNum++;
						fam = fam + "g ";
						n = true;
					}
				}
			}
			famSize = boyNum + girlNum;
			if (famSize <= 2) {
				TwoChildren++;
				sum = sum + 2;
			} else if (famSize == 3) {
				ThreeChildren++;
				sum = sum + 3;
			} else {
				FourChildren++;
				sum = sum + famSize;
			}
		}
			System.out.println("Average: " + (sum/Z) + " children to get at least
one of each gender.");
			System.out.println("Number of families with 2 children: " +
TwoChildren);
```

```
				System.out.println("Number of families with 3 children: " +
ThreeChildren);
				System.out.println("Number of families with 4 or more children: " +
FourChildren);
				String most = (FourChildren > Math.max(TwoChildren,
ThreeChildren)) ? "4 or more" : ((ThreeChildren > TwoChildren) ? "3" : "2");
				System.out.println("The most common number of children is " +
most + ".");
		}
}
```