Divisors

```
/**
 *  Gets a command-line argument (int), and prints all the divisors of the given
number.
 */
public class Divisors {
	public static void main (String[] args) {
		// given a number x, print all its divisors
		int num = Integer.parseInt(args[0]);

		// i cannot be zero because then we'll be dividing by zero which is
undefined. We'll start from 1 and end the count with the number given.
		for (int i = 1; i <= num; i++) {
			if (num % i == 0) {
				System.out.println(i);
			}
		}
	}
}
```

Reverse

```
/**
 * 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 str = args[0];

		int j = str.length();
		int middle = j%2==0 ? (j-1) / 2 : j / 2;
		String reversed = "";

		while (j != 0) {
			reversed += str.charAt(j - 1);
			j -= 1;
		}

		System.out.println(reversed + "\nThe middle character is " +
str.charAt(middle));
	}
}
```

InOrder

```
/**
 *  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 random, followingRand;

                random = (int) (10 * Math.random());

                do {
                        followingRand = random;
                        System.out.println(followingRand);
                        random = (int) (10 * Math.random());

                } while (followingRand < random);
        }
}
```

Perfect Numbers

```
/**
 *  Gets a command-line argument (int), and chekcs if the given number is perfect.
 */
public class Perfect {
	public static void main (String[] args) {
		int input = Integer.parseInt(args[0]);
		int i = 1; // We'll start from 1 so we won't divide by 0
		int sum = 1; // We'll start from 1 because 1 is always a divider of
number n, n >= 1
		String divisors = "1"; // 1 + ...other dividers (if any)

		while (i < input) {
			i++;
			if (input % i == 0 && i != input) {
				sum += i;
				divisors += " + " + i;
			}
		}

		if (sum == input) {
			System.out.print(input + " is a perfect number since " + input +
" = " + divisors);
		} else {
			System.out.println(input + " is not a perfect number"); // 6, 24,
28, 496, 5002, 8128
		}
	}
}
```

DamkaBoard

```
/**
 *  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 lines = Integer.parseInt(args[0]);

		for (int i = 0; i < lines; i++) {
			for (int j = 0; j < lines; j++) {
				if (i % 2 == 0) {
					System.out.print("* ");
				}
				else {
					System.out.print(" *");
				}
			}
			if (i != lines-1) {
				System.out.println("");
			}
		}
	}
}
```

OneOfEach

```
/**
 *  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) {
		double child;
		int count = 0;

		boolean girl = false;
		boolean boy = false;

		while (!(girl && boy)) {
			count++;
			child = Math.random();
			if (child > 0.5) {
				girl = true;
				System.out.print("g ");
			} else {
				boy = true;
				System.out.print("b ");
			}
		}

		System.out.println("\nYou made it... and you now have " + count + "
children.");
	}
}
```

OneOfEachStats1

```
/**
 *  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) {
                double child;
                int count = 0;

                boolean girl = false;
                boolean boy = false;

                // New variables
                int experiments = Integer.parseInt(args[0]);
                double average;
                String mostCommon;
                int twoChildren = 0;
                int threeChildren = 0;
                int fourOrMoreChildren = 0;
                int totalChildren = 0;

                for (int i = 0; i < experiments; i++) {
                        count = 0;
                        girl = false;
                        boy = false;
                        while (!(girl && boy)) {
```

```
                child = Math.random();
                if (child > 0.5) {
                    girl = true;
                } else {
                    boy = true;
                }
                count++;
            }
            totalChildren += count;
            if (count == 2) {
                twoChildren += 1;
            }
            else if (count == 3) {
                threeChildren += 1;
            } else {
                fourOrMoreChildren += 1;
            }

        }
        average = Double.valueOf(totalChildren) /
Double.valueOf(experiments);

        if (twoChildren > threeChildren && twoChildren >
fourOrMoreChildren) {
            mostCommon = "2";
        }
        else if (threeChildren > twoChildren && threeChildren >
fourOrMoreChildren) {
            mostCommon = "3";
        }
```

```
		else {
			mostCommon = "4 or more";
		}


		System.out.println("Average: " + average + " 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 children: " +
fourOrMoreChildren);
		System.out.println("The most common number of children is " +
mostCommon + ".");
	}
}
```

OneOfEachStats

```
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 child;
                int count = 0;

                boolean girl = false;
                boolean boy = false;

                // New variables
                double average;
                String mostCommon;
                int twoChildren = 0;
                int threeChildren = 0;
int fourOrMoreChildren = 0;

int totalChildren = 0;


for (int i = 0; i < T; i++) {

        count = 0;

        girl = false;

        boy = false;

        while (!(girl && boy)) {

                child = generator.nextDouble();

                if (child > 0.5) {

                        girl = true;

                } else {

                        boy = true;

                }

                count++;

        }

        totalChildren += count;

        if (count == 2) {

                twoChildren += 1;

        }

        else if (count == 3) {

                threeChildren += 1;

        } else {

                fourOrMoreChildren += 1;

        }


}

average = Double.valueOf(totalChildren) / Double.valueOf(T);
```

```
		if (twoChildren > threeChildren && twoChildren > 
fourOrMoreChildren) {
			mostCommon = "2";
		}
		else if (threeChildren > twoChildren && threeChildren > 
fourOrMoreChildren) {
			mostCommon = "3";
		}
		else {
			mostCommon = "4 or more";
		}

		System.out.println("Average: " + average + " children to get at least 
one of each gender.\nNumber of families with 2 children: " + 
							twoChildren + "\nNumber of 
families with 3 children: " + threeChildren + "\nNumber of families with 4 or more 
children: " + 
							fourOrMoreChildren + "\nThe 
most common number of children is " + mostCommon + ".");

	}
}
```