1 Exercises

Exercise 1. Consider the following method:

```java
public static int mystery(int[] a) {
    int x = 0;
    for (int i = 0; i < a.length; i++) {
        x += a[i] * a[i];
    }
    return x;
}
```

a. What does `mystery()` compute and return in general?
b. What will `mystery()` return if the argument `a` is an integer array containing the values 1, 2, 3, 4, and 5?

Exercise 2. Consider the following method:

```java
public static int[][] mystery(int n, int k) {
    int[][] a = new int[n][n];
    for (int i = 0; i < n; i++) {
        for (int j = 0; j < n; j++) {
            a[i][j] = (i > j) ? 0 : k;
        }
    }
    return a;
}
```

a. What does `mystery()` compute and return in general?
b. What will `mystery()` return if the arguments are `n = 4` and `k = 5`?

Exercise 3. Consider the following recursive method:

```java
public static int mystery(int a, int b) {
    if (b == 0) {
        return 0;
    }
    if (a == 0) {
        return mystery(b - 1, a);
    }
    return b + mystery(b, a - 1);
}
```

a. What will `mystery(0, 10)` return?
b. What will `mystery(10, 3)` return?
c. What will `mystery(200, 300)` return?
d. What does `mystery(a, b)` return in general about `a` and `b`?

Exercise 4. Consider the following program:

```java
public class Circle {
    public static void main(String[] args) {
        double r = Double.parseDouble(args[0]);
        double c = 2 * Math.PI * r;
        double a = Math.PI * r * r;
        StdOut.printf("radius = %.2f, circumference = %.2f, area = %.2f\n", r, c, a);
    }
}
```
a. What does circle compute and write in general?

b. What will circle write when run with the command-line argument 5?

**Exercise 5.** Write a program called RandomInts.java that accepts \( n \) (int), \( a \) (int), and \( b \) (int) as command-line arguments and writes \( n \) random integers (one per line) from the range \([a, b]\). For example,

```bash
~/workspace/dsa/programs $ java RandomInts 5 100 1000
257
197
670
446
590
```

**Exercise 6.** Write a program called Stats.java that reads integers from standard input, and computes and writes their mean, variance, and standard deviation, each up to 3 decimal places. For example,

```bash
~/workspace/dsa/programs $ java Stats
1 2 3 4 5 <ctrl-d>
mean = 3.000 , var = 2.000 , std = 1.414
```

**Exercise 7.**

a. What is the command for generating 10000 random integers from the interval \([1, 24]\) using the RandomInts program from Problem 5?

b. What is the command for generating 10000 random integers from the interval \([1, 24]\) and saving the output in a file called numbers.txt?

c. What is the command for using the Stats program from Problem 6 to calculate stats for the numbers in numbers.txt?

d. What is the command to perform the last two tasks in one shot?

## 2 Solutions

**Solution 1.**

a. The sum of squares of the integers in \( a \).

b. 55

**Solution 2.**

a. An \( n \)-by-\( n \) matrix in which the elements below the main diagonal are zeros and the rest of the elements are \( k \).

b. \( \{(5, 5, 5, 5), (0, 5, 5, 5), (0, 0, 5, 5), (0, 0, 0, 5)\} \)

**Solution 3.**

a. 0

b. 30

c. 60000
Solution 4.

a. Accepts $r$ (double) as command-line argument, computes the circumference $c$ and area $a$ of a circle with radius $r$, and writes the values of $r$, $c$, and $a$ up to 2 decimal places.

b. radius = 5.00, circumference = 31.42, area = 78.54

Solution 5.

```java
import stdlib.StdOut;
import stdlib.StdRandom;

public class RandomInts {
    public static void main(String[] args) {
        int n = Integer.parseInt(args[0]);
        int a = Integer.parseInt(args[1]);
        int b = Integer.parseInt(args[2]);
        for (int i = 0; i < n; i++) {
            int r = StdRandom.uniform(a, b + 1);
            StdOut.println(r);
        }
    }
}
```

Solution 6.

```java
import stdlib.StdIn;
import stdlib.StdOut;

public class Stats {
    public static void main(String[] args) {
        int a[] = StdIn.readInts();
        double mean = 0.0;
        double var = 0.0;
        double std = 0.0;
        for (int i = 0; i < a.length; i++) {
            mean += a[i];
        }
        mean /= a.length;
        for (int i = 0; i < a.length; i++) {
            var += (a[i] - mean) * (a[i] - mean);
        }
        var /= a.length;
        std = Math.pow(var, 0.5);
        StdOut.printf("mean = %.3f, var = %.3f, std = %.3f\n", mean, var, std);
    }
}
```

Solution 7.

a. java RandomInts 10000 1 24

b. java RandomInts 10000 1 24 > numbers.txt

c. java Stats < numbers.txt

d. java RandomInts 10000 1 24 | java Stats