Programming Exercises

Chapter 3: Selections

***3.1** (*Algebra: solve quadratic equations*) The two roots of a quadratic equation $ax^2 + bx + c = 0$ can be obtained using the following formula:

$$r_1 = \frac{-b + \sqrt{b^2 - 4ac}}{2a} \quad \text{and} \quad r_2 = \frac{-b - \sqrt{b^2 - 4ac}}{2a}$$

$b^2 - 4ac$ is called the discriminant of the quadratic equation. If it is positive, the equation has two real roots. If it is zero, the equation has one root. If it is negative, the equation has no real roots.

Write a program that prompts the user to enter values for a, b, and c and displays the result based on the discriminant. If the discriminant is positive, display two roots. If the discriminant is 0, display one root. Otherwise, display "The equation has no real roots".

Note that you can use `Math.pow(x, 0.5)` to compute $\sqrt{x}$. Here are some sample runs.

```
Enter a, b, c: 1.0 3 1 Enter
The equation has two roots -0.381966 and -2.61803
```

```
Enter a, b, c: 1 2.0 1 Enter
The equation has one root -1
```

```
Enter a, b, c: 1 2 3 Enter
The equation has no real roots
```

***3.5** (*Find future dates*) Write a program that prompts the user to enter an integer for today's day of the week (Sunday is 0, Monday is 1, …, and Saturday is 6). Also prompt the user to enter the number of days after today for a future day and display the future day of the week. Here is a sample run:

```
Enter today's day: 1 Enter
Enter the number of days elapsed since today: 3 Enter
Today is Monday and the future day is Thursday
```

```
Enter today's day: 0 Enter
Enter the number of days elapsed since today: 31 Enter
Today is Sunday and the future day is Wednesday
```

***3.11** (*Find the number of days in a month*) Write a program that prompts the user to enter the month and year and displays the number of days in the month. For example, if the user entered month **2** and year **2012**, the program should display that February 2012 had 29 days. If the user entered month **3** and year **2015**, the program should display that March 2015 had 31 days.

***3.17** (*Game: scissor, rock, paper*) Write a program that plays the popular scissor-rock-paper game. (A scissor can cut a paper, a rock can knock a scissor, and a paper can wrap a rock.) The program randomly generates a number **0**, **1**, or **2** representing scissor, rock, and paper. The program prompts the user to enter a number **0**, **1**, or **2** and displays a message indicating whether the user or the computer wins, loses, or draws. Here are sample runs:

```
scissor (0), rock (1), paper (2): 1 Enter
The computer is scissor. You are rock. You won
```

```
scissor (0), rock (1), paper (2): 2 Enter
The computer is paper. You are paper too. It is a draw
```

***3.31** (*Financials: currency exchange*) Write a program that prompts the user to enter the exchange rate from currency in U.S. dollars to Chinese RMB. Prompt the user to enter **0** to convert from U.S. dollars to Chinese RMB and **1** to convert from Chinese RMB and U.S. dollars. Prompt the user to enter the amount in U.S. dollars or Chinese RMB to convert it to Chinese RMB or U.S. dollars, respectively. Here are the sample runs:

```
Enter the exchange rate from dollars to RMB: 6.81 Enter
Enter 0 to convert dollars to RMB and 1 vice versa: 0 Enter
Enter the dollar amount:  100 Enter
$100.0 is 681.0 yuan
```

```
Enter the exchange rate from dollars to RMB: 6.81 Enter
Enter 0 to convert dollars to RMB and 1 vice versa: 5 Enter
Enter the RMB amount:  10000 Enter
10000.0 yuan is $1468.43
```

```
Enter the exchange rate from dollars to RMB: 6.81 Enter
Enter 0 to convert dollars to RMB and 1 vice versa: 5 Enter
Incorrect input
```

***3.33** (*Financial: compare costs*) Suppose you shop for rice in two different packages. You would like to write a program to compare the cost. The program prompts the user to enter the weight and price of the each package and displays the one with the better price. Here is a sample run:

```
Enter weight and price for package 1: 50 24.59 Enter
Enter weight and price for package 2: 25 11.99 Enter
Package 2 has a better price.
```

```
Enter weight and price for package 1: 50 25 ↵ Enter
Enter weight and price for package 2: 25 12.5 ↵ Enter
Two packages have the same price.
```

Chapter 5: Loops

***5.1** (*Count positive and negative numbers and compute the average of numbers*) Write a program that reads an unspecified number of integers, determines how many positive and negative values have been read, and computes the total and average of the input values (not counting zeros). Your program ends with the input **0**. Display the average as a floating-point number. Here is a sample run:

```
Enter an integer, the input ends if it is 0: 1 2 -1 3 0 ↵ Enter
The number of positives is 3
The number of negatives is 1
The total is 5.0
The average is 1.25
```

```
Enter an integer, the input ends if it is 0: 0 ↵ Enter
No numbers are entered except 0
```

5.3 (*Conversion from kilograms to pounds*) Write a program that displays the following table (note that **1** kilogram is **2.2** pounds):

```
Kilograms      Pounds
1                2.2
3                6.6
...
197            433.4
199            437.8
```

5.5 (*Conversion from kilograms to pounds and pounds to kilograms*) Write a program that displays the following two tables side by side:

```
Kilograms   Pounds     |      Pounds      Kilograms
1             2.2     |      20              9.09
3             6.6     |      25             11.36
...
197         433.4     |      510           231.82
199         437.8     |      515           234.09
```

***5.9** (*Find the two highest scores*) Write a program that prompts the user to enter the number of students and each student's name and score, and finally displays the student with the highest score and the student with the second-highest score.

5.11 (*Find numbers divisible by 5 or 6, but not both*) Write a program that displays all the numbers from 100 to 200, ten per line, that are divisible by 5 or 6, but not both. Numbers are separated by exactly one space.

5.13 (*Find the largest* `n` such that $n^3 < 12{,}000$) Use a `while` loop to find the largest integer `n` such that n^3 is less than 12,000.

***5.15** (*Display the ASCII character table*) Write a program that prints the characters in the ASCII character table from `!` to `~`. Display ten characters per line. The ASCII table is shown in Appendix B. Characters are separated by exactly one space.

***5.23** (*Demonstrate cancellation errors*) A cancellation error occurs when you are manipulating a very large number with a very small number. The large number may cancel out the smaller number. For example, the result of `100000000.0 + 0.000000001` is equal to `100000000.0`. To avoid cancellation errors and obtain more accurate results, carefully select the order of computation. For example, in computing the following series, you will obtain more accurate results by computing from right to left rather than from left to right:

$$1 + \frac{1}{2} + \frac{1}{3} + \ldots + \frac{1}{n}$$

Write a program that compares the results of the summation of the preceding series, computing from left to right and from right to left with `n = 50000`.

***5.31** (*Financial application: compute CD value*) Suppose you put $10,000 into a CD with an annual percentage yield of 5.75%. After one month, the CD is worth

```
10000 + 10000 * 5.75 / 1200 = 10047.92
```

After two months, the CD is worth

```
10047.91 + 10047.91 * 5.75 / 1200 = 10096.06
```

After three months, the CD is worth

```
10096.06 + 10096.06 * 5.75 / 1200 = 10144.44
```

and so on.

Write a program that prompts the user to enter an amount (e.g., `10000`), the annual percentage yield (e.g., `5.75`), and the number of months (e.g., `18`) and displays a table as shown in the sample run.

***5.35** (*Summation*) Write a program to compute the following summation.

$$\frac{1}{1 + \sqrt{2}} + \frac{1}{\sqrt{2} + \sqrt{3}} + \frac{1}{\sqrt{3} + \sqrt{4}} + \ldots + \frac{1}{\sqrt{624} + \sqrt{625}}$$

***5.39** (*Financial application: find the sales amount*) You have just started a sales job in a department store. Your pay consists of a base salary and a commission. The base salary is $5,000. The scheme shown below is used to determine the commission rate.

Sales Amount	Commission Rate
$0.01–$5,000	8 percent
$5,000.01–$10,000	10 percent
$10,000.01 and above	12 percent

Note that this is a graduated rate. The rate for the first $5,000 is at 8%, the next $5000 is at 10%, and the rest is at 12%. If the sales amount is 25,000, the commission is 5,000 * 8% + 5,000 * 10% + 15,000 * 12% = 2,700.

Your goal is to earn $30,000 a year. Write a program that finds the minimum sales you have to generate in order to make $30,000.

***5.41** (*Occurrence of max numbers*) Write a program that reads integers, finds the largest of them, and counts its occurrences. Assume that the input ends with number `0`. Suppose that you entered `3 5 2 5 5 5 0`; the program finds that the largest is `5` and the occurrence count for `5` is `4`.

(*Hint*: Maintain two variables, `max` and `count`. `max` stores the current max number, and `count` stores its occurrences. Initially, assign the first number to `max` and `1` to `count`. Compare each subsequent number with `max`. If the number is greater than `max`, assign it to `max` and reset `count` to `1`. If the number is equal to `max`, increment `count` by `1`.)

```
Enter numbers: 3 5 2 5 5 5 0 ↵Enter
The largest number is 5
The occurrence count of the largest number is 4
```

***5.43** (*Math: combinations*) Write a program that displays all possible combinations for picking two numbers from integers 1 to 7. Also display the total number of all combinations.

```
1 2
1 3
...
...

The total number of all combinations is 21
```

***5.47** (*Business: check ISBN-13*) **ISBN-13** is a new standard for indentifying books. It uses 13 digits $d_1d_2d_3d_4d_5d_6d_7d_8d_9d_{10}d_{11}d_{12}d_{13}$. The last digit d_{13} is a checksum, which is calculated from the other digits using the following formula:

$$10 - (d_1 + 3d_2 + d_3 + 3d_4 + d_5 + 3d_6 + d_7 + 3d_8 + d_9 + 3d_{10} + d_{11} + 3d_{12})\%10$$

If the checksum is 10, replace it with 0. Your program should read the input as a string. Here are sample runs:

```
Enter the first 12 digits of an ISBN-13 as a string: 978013213080 Enter
The ISBN-13 number is 9780132130806
```

```
Enter the first 12 digits of an ISBN-13 as a string: 978013213079 Enter
The ISBN-13 number is 9780132130790
```

```
Enter the first 12 digits of an ISBN-13 as a string: 97801320 Enter
97801320 is an invalid input
```

***5.49** (*Count vowels and consonants*) Assume letters `A`, `E`, `I`, `O`, and `U` as the vowels. Write a program that prompts the user to enter a string and displays the number of vowels and consonants in the string.

```
Enter a string: Programming is fun ↵ Enter
The number of vowels is 5
The number of consonants is 11
```

***5.51** (*Longest common prefix*) Write a program that prompts the user to enter two strings and displays the largest common prefix of the two strings. Here are some sample runs:

```
Enter the first string: Welcome to C++ ↵ Enter
Enter the second string: Welcome to programming ↵ Enter
The common prefix is Welcome to
```

```
Enter the first string: Atlanta ↵ Enter
Enter the second string: Macon ↵ Enter
Atlanta and Macon have no common prefix
```

***7.1** (*Assign grades*) Write a program that reads student scores, gets the best score, and then assigns grades based on the following scheme:

Grade is A if score is $\geq$ best $-$ 10

Grade is B if score is $\geq$ best $-$ 20;

Grade is C if score is $\geq$ best $-$ 30;

Grade is D if score is $\geq$ best $-$ 40;

Grade is F otherwise.

The program prompts the user to enter the total number of students, then prompts the user to enter all of the scores, and concludes by displaying the grades. Here is a sample run:

```
Enter the number of students: 4 Enter
Enter 4 scores: 40 55 70 58 Enter
Student 0 score is 40 and grade is C
Student 1 score is 55 and grade is B
Student 2 score is 70 and grade is A
Student 3 score is 58 and grade is B
```

***7.7** (*Count single digits*) Write a program that generates 100 random integers between 0 and 9 and displays the count for each number. (*Hint*: Use an array of ten integers, say `counts`, to store the counts for the number of 0s, 1s, ..., 9s.)

7.9 (*Find the smallest element*) Write a method that finds the smallest element in an array of double values using the following header:

```
public static double min(double[] array)
```

Write a test program that prompts the user to enter ten numbers, invokes this method to return the minimum value, and displays the minimum value. Here is a sample run of the program:

```
Enter ten numbers: 1.9 2.5 3.7 2 1.5 6 3 4 5 2 Enter
The minimum number is: 1.5
```

***7.11** (*Statistics: compute deviation*) Programming Exercise 5.45 computes the standard deviation of numbers. This exercise uses a different but equivalent formula to compute the standard deviation of **n** numbers.

$$mean = \frac{\sum_{i=1}^{n} x_i}{n} = \frac{x_1 + x_2 + \cdots + x_n}{n} \quad deviation = \sqrt{\frac{\sum_{i=1}^{n} (x_i - mean)^2}{n - 1}}$$

To compute the standard deviation with this formula, you have to store the individual numbers using an array, so that they can be used after the mean is obtained.

Your program should contain the following methods:

```
/** Compute the deviation of double values */
public static double deviation(double[] x)

/** Compute the mean of an array of double values */
public static double mean(double[] x)
```

Write a test program that prompts the user to enter ten numbers and displays the mean and standard deviation, as shown in the following sample run:

```
Enter ten numbers: 1.9 2.5 3.7 2 1 6 3 4 5 2 Enter
The mean is 3.11
The standard deviation is 1.55738
```

***7.13** (*Random number chooser*) Write a method that returns a random number between 1 and 54, excluding the numbers passed in the argument. The method header is specified as follows:

```
public static int getRandom(int... numbers)
```

7.15 (*Eliminate duplicates*) Write a method that returns a new array by eliminating the duplicate values in the array using the following method header:

```
public static int[] eliminateDuplicates(int[] list)
```

Write a test program that reads in ten integers, invokes the method, and displays the result. Here is the sample run of the program:

```
Enter ten numbers: 1 2 3 2 1 6 3 4 5 2 Enter
The distinct numbers are: 1 2 3 6 4 5
```

7.27 (*Identical arrays*) The arrays `list1` and `list2` are *identical* if they have the same contents. Write a method that returns `true` if `list1` and `list2` are identical, using the following header:

```
public static boolean equals(int[] list1, int[] list2)
```

Write a test program that prompts the user to enter two lists of integers and displays whether the two are identical. Here are the sample runs. Note that the first number in the input indicates the number of the elements in the list. This number is not part of the list.

```
Enter list1: 5 2 5 6 6 1 Enter
Enter list2: 5 5 2 6 1 6 Enter
Two lists are identical
```

```
Enter list1: 5 5 5 6 6 1 Enter
Enter list2: 5 2 5 6 1 6 Enter
Two lists are not identical
```

***7.29** (*Game: pick four cards*) Write a program that picks four cards from a deck of 52 cards and computes their sum. An Ace, King, Queen, and Jack represent 1, 13, 12, and 11, respectively. Your program should display the number of picks that yields the sum of 24.

---

no star = easy
* = moderate
** = hard
*** = challenging

From Introduction to Java Programming, Comprehensive Version (10th Edition) by Y. Daniel Liang