Labs

Lab 3.1: Prime Numbers

What is the purpose?

In this lab exercise, you will rewrite Listing 4.14, PrimeNumber.java, on pages 125-126 of your textbook.

Your program must meet the following requirements:

1. Declare a method to determine whether an integer is a prime number. Use the following method header:

   ```
   public static Boolean isPrime(int num)
   ```

   An integer greater than 1 is a prime number if its only divisor is 1 or itself. For example, isPrime(11) returns true, and isPrime(9) returns false.

2. Use the isPrime method to find the first 1,000 prime numbers and display every 10 prime numbers in a row.

What are the steps?

- Task 1:

  Procedure

  1. Study the sample program on page 117 of your textbook.
  2. Declare a method called isPrime(int).
  3. Use a for loop to determine if a number (besides 1) is prime or not.
  4. Add another for loop to print out a row of 10 prime numbers for the first 1000 prime numbers.
  5. Compile the java file using the javac command.
  6. Execute the java class using the java command.
  7. Save a screen shot of the output and submit it to your instructor. It should look similar to Figure 3-1-1.

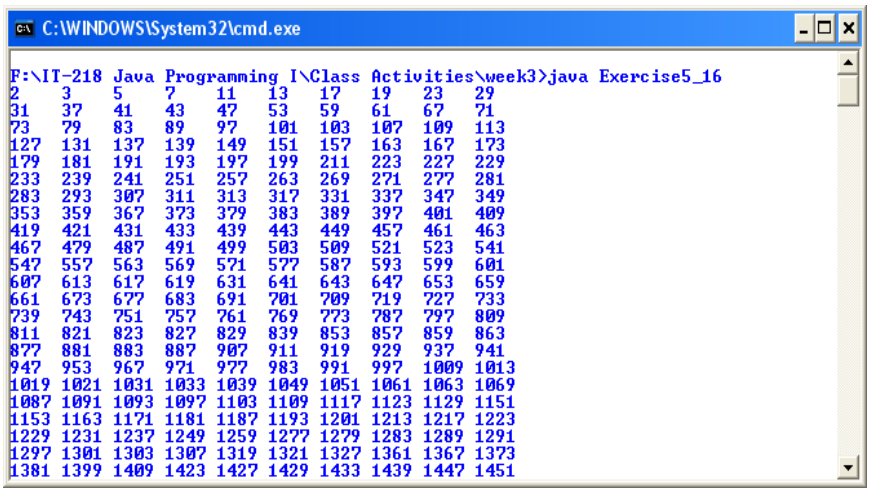

Figure 3-1-1

Did it work?

- Were you able to display the first 1000 prime numbers in the console?
- Were you able to display every 10 prime numbers in a row in the console?

Lab 3.2: Analyzing Scores

What is the purpose?

In this lab, you will write a program that reads an unspecified number of scores by an input dialog box and determines how many scores are above or equal to the average and how many scores are below the average. Enter a negative number to signify the end of the input. Assume that the maximum number of scores is 100. Display the output in the console.

What are the steps?

- Task 1:

  Procedure

  1. Create a Java class and name the java file with .java extension.
  2. Import the javax.swing.JOptionPane package to create dialog boxes.
  3. Declare a double array to capture all the input scores.
  4. Declare a double variable to store the summation of all scores.
  5. Create a do-while loop to process all the scores and the summation.
  6. Declare an integer variable to count the total number of scores.
  7. Declare a double variable to compute the average of all scores.
  8. Declare an integer variable to count the number of scores above the average.

9. Declare an integer variable to count the number of scores below the average.
10. Create a for loop to compute scores' average, the number of scores above the average, and the number of scores below the average.
11. Compile the java file using the javac command.
12. Execute the java class using the java command.
13. Save screen shots of the output similar to Figures 3-2-1 through 3-2-6 and submit to them to your instructor.

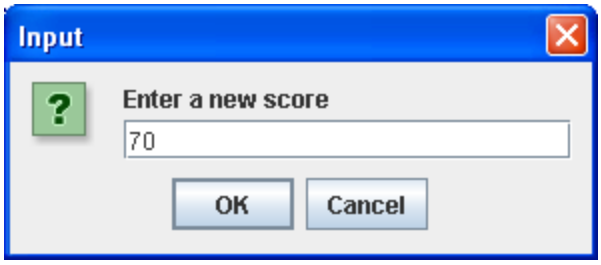

Figure 3-2-1: Sample Output 1

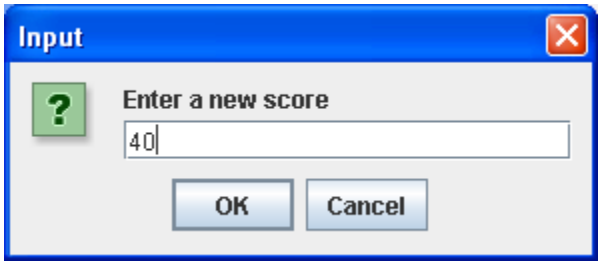

Figure 3-2-2: Sample Output 2

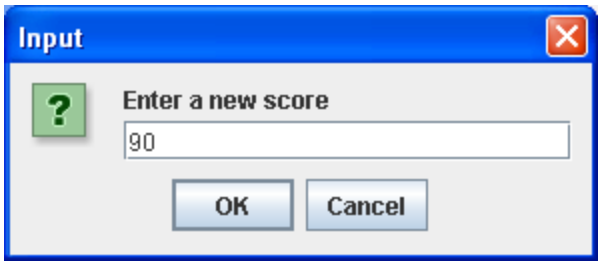

Figure 3-2-3: Sample Output 3

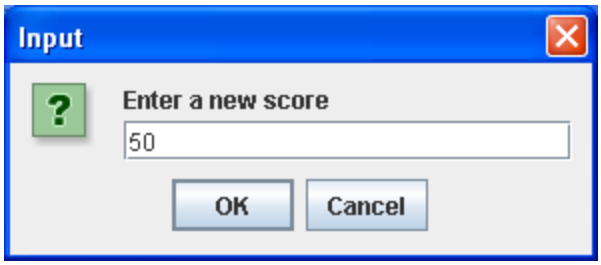

Figure 3-2-4: Sample Output 4

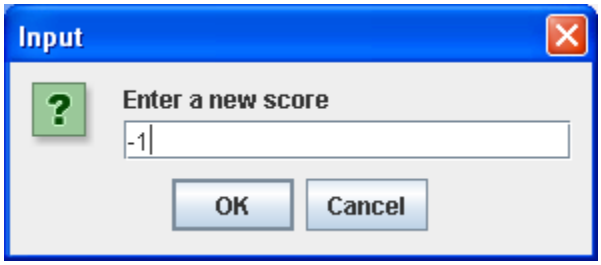

Figure 3-2-5: Sample Output 5

```
Average is 62.25
The number of scores above or equal to the average 2
The number of scores below the average 2
```

Figure 3-2-6: Console Output

Did it work?

Were you able to—

- Capture undefined number of scores and display the average score?
- Display the number of scores are above the average score?
- Display the number of scores are below the average score?