# JAVA PROGRAMMING PRACTICE EXERCISES 2

|  |  |
| --- | --- |
| **Exercise 1** | Write a program that displays the dimensions of a letter-size (8.5 × 11 inches) sheet of paper in millimeters. There are 25.4 millimeters per inch. Use constants and comments in your program. |

|  |  |
| --- | --- |
| **Exercise 2** | Write a program that computes and displays the perimeter of a letter-size (8.5 × 11 inches) sheet of paper and the length of its diagonal. |

|  |  |
| --- | --- |
| **Exercise 3** | Write a program that reads a number and displays the square, cube, and fourth power. Use the **Math.pow** method only for the fourth power. |

|  |  |
| --- | --- |
| **Exercise 4** | Write a program that prompts the user for two integers and then prints:   * The sum * The difference * The product * The average * The distance (absolute value of the difference) * The maximum (the larger of the two) * The minimum (the smaller of the two)   ***Hint:***The max and min functions are declared in the **Math** class. |

|  |  |
| --- | --- |
| **Exercise 5** | Write a program that prompts the user for a measurement in metres and then converts it to miles, feet and inches. |

|  |  |
| --- | --- |
| **Exercise 6** | Write a program that prompts the user for a radius and then prints:   * The area and circumference of a circle with that radius * The volume and surface area of a sphere with that radius |

|  |  |
| --- | --- |
| **Exercise 7** | Write a program that helps a person decide whether to buy a car. Your program’s inputs should be:   * The cost of a new car * The estimated miles driven per year * The estimated petrol price * The efficiency in miles per gallon * The estimated resale value after 5 years   Compute the total cost of owning the car for five years. (For simplicity, we will not take the cost of financing into account.) Obtain realistic prices for a new and used car from the Internet. Run your program using today’s petrol price and 15,000 miles per year. |

|  |  |
| --- | --- |
| **Exercise 8** | Write a program that asks the user to input:   * The number of gallons of gas in the tank * The fuel efficiency in miles per gallon * The price of gas per gallon   Then print the cost per 100 miles and how far the car can go with the gas in the tank. |

|  |  |
| --- | --- |
| **Exercise 9** | **File names and extensions**.  Write a program that prompts the user for the drive letter (C), the path (\Windows\System), the file name (Readme), and the extension (txt). Then print the complete file name C:\Windows\System\Readme.txt. |

|  |  |
| --- | --- |
| **Exercise 10** | Write a program that reads a number between 1,000 and 999,999 from the user, where the user enters a comma in the input. Then print the number without a comma. Here is a sample dialog:  **Please enter an integer between 1,000 and 999,999**: 23,456  **23456**  **Hint**: Read the input as a string. Measure the length of the string. Suppose it contains n characters. Then extract substrings consisting of the first n – 4 characters and the last three characters. |

|  |  |
| --- | --- |
| **Exercise 11** | Write a program that reads a number between 1,000 and 999,999 from the user and prints it with a comma separating the thousands. Here is a sample dialog:  **Please enter an integer between 1000 and 999999**: 23456  **23,456** |

|  |  |
| --- | --- |
| **Exercise 12** | **Printing a grid.**  Write a program that prints the following grid to play tic-tac-toe.    Of course, you could simply write seven statements of the form  **System.out.println("+--+--+--+");**  You should do it the smart way, though. Declare string variables to hold two kinds of patterns: a comb-shaped pattern and the bottom line. Print the comb three times and the bottom line once. |

|  |  |
| --- | --- |
| **Exercise 13** | Write a program that reads in an integer and breaks it into a sequence of individual digits. For example, the input **16384** is displayed as  **1 6 3 8 4**  You may assume that the input has no more than five digits and is not negative. |

|  |  |
| --- | --- |
| **Exercise 14** | Write a program that reads two times in military format (0900, 1730) and prints the number of hours and minutes between the two times. Here is a sample run.  **Please enter the first time**: 0900  **Please enter the second time**: 1730  **8 hours 30 minutes**  Extra credit if you can deal with the case where the first time is later than the second:  **Please enter the first time**: 1730  **Please enter the second time**: 0900  **15 hours 30 minutes** |

|  |  |
| --- | --- |
| **Exercise 15** | **Writing large letters**.  A large letter H can be produced like this:    It can be declared as a string literal like this:  **final string LETTER\_H = "\* \*\n\* \*\n\*\*\*\*\*\n\* \*\n\* \*\n";**  (The \n escape sequence denotes a “newline” character that causes subsequent characters to be printed on a new line.)  Do the same for the letters E, L, and O. Then write the message:  **H**  **E**  **L**  **L**  **O**  in large letters. |

|  |  |
| --- | --- |
| **Exercise 16** | Write a program that transforms numbers **1, 2, 3, …, 12** into the corresponding month names **January, February, March, …, December**.  **Hint**: Make a very long string "January February March ...", in which you add spaces such that each month name has the same length. Then use substring to extract the month you want. |

|  |  |
| --- | --- |
| **Exercise 17** | Write a program that prints a Christmas tree:    Remember to use escape sequences. |

|  |  |
| --- | --- |
| **Exercise 18** | Easter Sunday is the first Sunday after the first full moon of spring. To compute the date, you can use this algorithm, invented by the mathematician Carl Friedrich Gauss in 1800:   1. Let **y** be the year (such as 1800 or 2001). 2. Divide **y** by **19** and call the remainder **a**. Ignore the quotient. 3. Divide **y** by **100** to get a quotient **b** and a remainder **c**. 4. Divide **b** by **4** to get a quotient **d** and a remainder **e**. 5. Divide **8 \* b + 13** by **25** to get a quotient **g**. Ignore the remainder. 6. Divide **19 \* a + b - d - g + 15** by **30** to get a remainder **h**. Ignore the quotient. 7. Divide **c** by **4** to get a quotient **j** and a remainder **k**. 8. Divide **a + 11 \* h** by **319** to get a quotient **m**. Ignore the remainder. 9. Divide **2 \* e + 2 \* j - k - h + m + 32** by **7** to get a remainder **r**. Ignore the quotient. 10. Divide **h - m + r + 90** by **25** to get a quotient **n**. Ignore the remainder. 11. Divide **h - m + r + n + 19** by **32** to get a remainder **p**. Ignore the quotient.   Then Easter falls on day **p** of month **n**. For example, if **y** is 2001:  **a = 6 h = 18 n = 4**  **b = 20, c = 1 j = 0, k = 1 p = 15**  **d = 5, e = 0 m = 0**  **g = 6 r = 6**  Therefore, in 2001, Easter Sunday fell on April 15. Write a program that prompts the user for a year and prints out the month and day of Easter Sunday. |

|  |  |
| --- | --- |
| **Exercise 19** | The following pseudocode describes how a bookstore computes the price of an order from the total price and the number of the books that were ordered.  **Read the total book price and the number of books.**  **Compute the tax (7.5 percent of the total book price).**  **Compute the shipping charge (£2 per book).**  **The price of the order is the sum of the total book price, the tax, and the shipping charge.**  **Print the price of the order.**  Translate this pseudocode into a Java program. |

|  |  |
| --- | --- |
| **Exercise 20** | The following pseudocode describes how to turn a string containing a ten-digit phone number (such as "4155551212") into a more readable string with parentheses and dashes, like this: "(415) 555-1212".  **Take the substring consisting of the first three characters and surround it with "(" and ") ". This is the area code.**  **Concatenate the area code, the substring consisting of the next three characters, a hyphen, and the substring consisting of the last four characters. This is the formatted number.**  Translate this pseudocode into a Java program that reads a telephone number into a string variable, computes the formatted number, and prints it. |

|  |  |
| --- | --- |
| **Exercise 21** | The following pseudocode describes how to extract the dollars and cents from a price given as a floating-point value. For example, a price **2.95** yields values **2** and **95** for the dollars and cents.  **Assign the price to an integer variable dollars.**  **Multiply the difference price - dollars by 100 and add 0.5.**  **Assign the result to an integer variable cents.**  Translate this pseudocode into a Java program. Read a price and print the dollars and cents. Test your program with inputs 2.95 and 4.35. |

|  |  |
| --- | --- |
| **Exercise 22** | An online bank wants you to create a program that shows prospective customers how their deposits will grow. Your program should read the initial balance and the annual interest rate. Interest is compounded monthly. Print out the balances after the first three months.  Here is a sample run:  **Initial balance: 1000**  **Annual interest rate in percent: 6.0**  **After first month: 1005.00**  **After second month: 1010.03**  **After third month: 1015.08** |

|  |  |
| --- | --- |
| **Exercise 23** | A video club wants to reward its best members with a discount based on the member’s number of movie rentals and the number of new members referred by the member. The discount is in percent and is equal to the sum of the rentals and the referrals, but it cannot exceed 75 percent. (**Hint**: **Math.min**.)  Write a program **Discount-Calculator** to calculate the value of the discount.  Here is a sample run:  **Enter the number of movie rentals: 56**  **Enter the number of members referred to the video club: 3**  **The discount is equal to: 59.00 percent**. |

|  |  |
| --- | --- |
| **Exercise 24** | Consider the following circuit.    Write a program that reads the resistances of the three resistors and computes the total resistance, using Ohm’s law. |

|  |  |
| --- | --- |
| **Exercise 25** | The dew point temperature Td can be calculated (approximately) from the relative humidity RH and the actual temperature T by    where **a** = **17.27** and **b** = **237.7° C**.  Write a program that reads the relative humidity (between 0 and 1) and the temperature (in degrees C) and prints the dew point value. Use the Java function **log** to compute the natural logarithm. |

|  |  |
| --- | --- |
| **Exercise 26** | The pipe clip temperature sensors shown here are robust sensors that can be clipped directly onto copper pipes to measure the temperature of the liquids in the pipes.    Each sensor contains a device called a thermistor. Thermistors are semiconductor devices that exhibit a temperature-dependent resistance described by:    where **R** is the resistance (in **Ω**) at the temperature **T** (in **°K**), and **R0** is the resistance (in Ω) at the temperature **T0** (in **°K**). **β** is a constant that depends on the material used to make the thermistor. Thermistors are specified by providing values for **R0**, **T0**, and **β.**  The thermistors used to make the pipe clip temperature sensors have  **R0 = 1075 Ω** at **T0 = 85 °C**, and **β = 3969 °K**.  (Notice that **β** has units of **°K**. Recall that the temperature in **°K** is obtained by adding **273** to the temperature in **°C**). The liquid temperature, in **°C**, is determined from the resistance **R**, in Ω, using    Write a Java program that prompts the user for the thermistor resistance **R** and prints a message giving the liquid temperature in **°C**. |