2

Introduction to Java Applications

What's in a name?
that which we call a rose
By any other name
would smell as sweet.
—William Shakespeare

When faced with a decision, I always ask, "What would be the most fun?"
—Peggy Walker

"Take some more tea," the March Hare said to Alice, very earnestly. "I've had nothing yet, "Alice replied in an offended tone: "so I can't take more." "You mean you can't take less," said the Hatter: "it's very easy to take more than nothing."
—Lewis Carroll

OBJECTIVES

In this chapter you will learn:

- To write simple Java applications.
- To use input and output statements.
- Java's primitive types.
- Basic memory concepts.
- To use arithmetic operators.
- The precedence of arithmetic operators.
- To write decision-making statements.
- To use relational and equality operators.

Student Solution Exercises

2.7 Fill in the blanks in each of the following statements:

a) _____ are used to document a program and improve its readability.

ANS: Comments.

b) A decision can be made in a Java program with a(n) _____.

ANS: `if` statement.

c) Calculations are normally performed by _____ statements.

ANS: assignment statements.

d) The arithmetic operators with the same precedence as multiplication are _____ and _____.

ANS: division (/), remainder (%)

e) When parentheses in an arithmetic expression are nested, the _____ set of parentheses is evaluated first.

ANS: innermost.

f) A location in the computer's memory that may contain different values at various times throughout the execution of a program is called a(n) _____.

ANS: variable.

2.9 State whether each of the following is *true* or *false*. If *false*, explain why.

a) Java operators are evaluated from left to right.

ANS: False. Some operators (e.g., assignment, =) evaluate from right to left.

b) The following are all valid variable names: `_under_bar_`, `m928134`, `t5`, `j7`, `her_sales$`, `his_$account_total`, `a`, `b$`, `c`, `z` and `z2`.

ANS: True.

c) A valid Java arithmetic expression with no parentheses is evaluated from left to right.

ANS: False. The expression is evaluated according to operator precedence.

d) The following are all invalid variable names: `3g`, `87`, `67h2`, `h22` and `2h`.

ANS: False. Identifier `h22` is a valid variable name.

2.11 Which of the following Java statements contain variables whose values are modified?

a) `p = i + j + k + 7;`

b) `System.out.println( "variables whose values are destroyed" );`

c) `System.out.println( "a = 5" );`

d) `value = input.nextInt();`

ANS: (a), (d).

2.12 Given that $y = ax^3 + 7$, which of the following are correct Java statements for this equation?

a) `y = a * x * x * x + 7;`

b) `y = a * x * x * ( x + 7 );`

c) `y = ( a * x ) * x * ( x + 7 );`

d) `y = ( a * x ) * x * x + 7;`

e) `y = a * ( x * x * x ) + 7;`

f) `y = a * x * ( x * x + 7 );`

ANS: (a), (d), (e)

2.14 Write an application that displays the numbers 1 to 4 on the same line, with each pair of adjacent numbers separated by one space. Write the program using the following techniques:

a) Use one `System.out.println` statement.
b) Use four `System.out.print` statements.
c) Use one `System.out.printf` statement.

ANS:

```
1  // Exercise 2.14 Solution: Printing.java
2  // Prints the numbers 1 through 4 several ways.
3
4  public class Printing
5  {
6     public static void main( String args[] )
7     {
8        System.out.print( "Part (a): " );
9
10       // one System.out.println statement
11       System.out.println( "1 2 3 4" );
12
13       System.out.print( "Part (b): " );
14
15       // four System.out.print statements
16       System.out.print( "1 " );
17       System.out.print( "2 " );
18       System.out.print( "3 " );
19       System.out.print( "4\n" );
20
21       System.out.print( "Part (c): " );
22
23       // one System.out.printf statement
24       System.out.printf( "%d %d %d %d\n", 1, 2, 3, 4 );
25    } // end main
26 } // end class Printing
```

```
Part (a): 1 2 3 4
Part (b): 1 2 3 4
Part (c): 1 2 3 4
```

2.16 Write an application that asks the user to enter two integers, obtains them from the user and displays the larger number followed by the words "`is larger`". If the numbers are equal, print the message "`These numbers are equal.`" Use the techniques shown in Fig. 2.15.

ANS:

```
1  // Exercise 2.16 Solution: Larger.java
2  // Program that determines the larger of two numbers.
3  import java.util.Scanner;
4
5  public class Larger
6  {
7     public static void main( String args[] )
8     {
9        Scanner input = new Scanner( System.in );
10
11       int number1; // first number to compare
12       int number2; // second number to compare
13
14       System.out.print( "Enter first integer: " ); // prompt for input
15       number1 = input.nextInt(); // read first number
16
17       System.out.print( "Enter second integer: " ); // prompt for input
18       number2 = input.nextInt(); // read second number
19
20       if ( number1 > number2 )
21          System.out.printf( "%d is larger\n", number1 );
22
23       if ( number1 < number2 )
24          System.out.printf( "%d is larger\n", number2 );
25
26       if ( number1 == number2 )
27          System.out.println( "These numbers are equal\n" );
28    } // end main
29 } // end class Larger
```

```
Enter first integer: 12
Enter second integer: 10
12 is larger
```

```
Enter first integer: 10
Enter second integer: 12
12 is larger
```

```
Enter first integer: 7
Enter second integer: 7
These numbers are equal
```

2.19 What does the following code print?

```
System.out.println( "*\n**\n***\n****\n*****" );
```

ANS:

```
*
**
***
****
*****
```

2.21 What does the following code print?

```
System.out.print( "*" );
System.out.print( "***" );
System.out.print( "*****" );
System.out.print( "****" );
System.out.println( "**" );
```

ANS:

```
***************
```

2.23 What does the following code print?

```
System.out.printf( "%s\n%s\n%s\n", "*", "***", "*****" );
```

ANS:

```
*
***
*****
```

2.25 Write an application that reads an integer and determines and prints whether it is odd or even. [*Hint*: Use the remainder operator. An even number is a multiple of 2. Any multiple of 2 leaves a remainder of 0 when divided by 2.]

ANS:

```
1  // Exercise 2.25 Solution: OddEven.java
2  // Program that determines if a number is odd or even.
3  import java.util.Scanner;
4
5  public class OddEven
6  {
7     public static void main( String args[] )
8     {
9        Scanner input = new Scanner( System.in );
10
11       int number; // number
```

```
12
13          System.out.print( "Enter integer: " ); // prompt for input
14          number = input.nextInt(); // read number
15
16          if ( number % 2 == 0 )
17             System.out.println( "Number is even" );
18
19          if ( number % 2 != 0 )
20             System.out.println( "Number is odd" );
21       } // end main
22    } // end class OddEven
```

```
Enter integer: 17
Number is odd
```

```
Enter integer: 244
Number is even
```

2.27 Write an application that displays a checkerboard pattern, as follows:

```
* * * * * * * *
 * * * * * * * *
* * * * * * * *
 * * * * * * * *
* * * * * * * *
 * * * * * * * *
* * * * * * * *
 * * * * * * * *
```

ANS:

```
1  // Exercise 2.27 Solution: Checker.java
2  // Program that draws a checkerboard.
3
4  public class Checker
5  {
6     public static void main( String args[] )
7     {
8        System.out.println( "* * * * * * * *" );
9        System.out.println( " * * * * * * * *" );
10       System.out.println( "* * * * * * * *" );
11       System.out.println( " * * * * * * * *" );
12       System.out.println( "* * * * * * * *" );
13       System.out.println( " * * * * * * * *" );
14       System.out.println( "* * * * * * * *" );
15       System.out.println( " * * * * * * * *" );
16    } // end main
17 } // end class Checker
```

```
* * * * * * * * *
 * * * * * * * * *
* * * * * * * * *
 * * * * * * * * *
* * * * * * * * *
 * * * * * * * * *
* * * * * * * * *
 * * * * * * * * *
```

2.28 Here's a peek ahead. In this chapter, you have learned about integers and the type `int`. Java can also represent floating-point numbers that contain decimal points, such as 3.14159. Write an application that inputs from the user the radius of a circle as an integer and prints the circle's diameter, circumference and area using the floating-point value 3.14159 for π. Use the techniques shown in Fig. 2.7. [*Note*: You may also use the predefined constant `Math.PI` for the value of π. This constant is more precise than the value 3.14159. Class `Math` is defined in package `java.lang`. Classes in that package are imported automatically, so you do not need to `import` class `Math` to use it.] Use the following formulas (*r* is the radius):

diameter = 2*r*
circumference = 2π*r*
area = πr^2

Do not store the results of each calculation in a variable. Rather, specify each calculation as the value that will be output in a `System.out.printf` statement. Note that the values produced by the circumference and area calculations are floating-point numbers. Such values can be output with the format specifier `%f` in a `System.out.printf` statement. You will learn more about floating-point numbers in Chapter 3.

ANS:

```
1  // Exercise 2.28 Solution: Circle.java
2  // Program that calculates area, circumference
3  // and diameter for a circle.
4  import java.util.Scanner;
5
6  public class Circle
7  {
8     public static void main( String args[] )
9     {
10       Scanner input = new Scanner( System.in );
11
12       int radius; // radius of circle
13
14       System.out.print( "Enter radius: " ); // prompt for input
15       radius = input.nextInt(); // read number
16
17       System.out.printf( "Diameter is %d\n", ( 2 * radius ) );
18       System.out.printf( "Area is %f\n", ( Math.PI * radius * radius ) );
19       System.out.printf( "Circumference is %f\n",
20          ( 2 * Math.PI * radius ) );
21    } // end main
22 } // end class Circle
```

```
Enter radius: 3
Diameter is 6
Area is 28.274334
Circumference is 18.849556
```

2.29 Here's another peek ahead. In this chapter, you have learned about integers and the type `int`. Java can also represent uppercase letters, lowercase letters and a considerable variety of special symbols. Every character has a corresponding integer representation. The set of characters a computer uses and the corresponding integer representations for those characters is called that computer's character set. You can indicate a character value in a program simply by enclosing that character in single quotes, as in `'A'`.

You can determine the integer equivalent of a character by preceding that character with `(int)`, as in

```
(int) 'A'
```

This form is called a cast operator. (You will learn about cast operators in Chapter 4.) The following statement outputs a character and its integer equivalent:

```
System.out.printf(
   "The character %c has the value %d\n", 'A', ( (int) 'A' ) );
```

When the preceding statement executes, it displays the character `A` and the value `65` (from the so-called Unicode® character set) as part of the string. Note that the format specifier `%c` is a placeholder for a character (in this case, the character `'A'`).

Using statements similar to the one shown earlier in this exercise, write an application that displays the integer equivalents of some uppercase letters, lowercase letters, digits and special symbols. Display the integer equivalents of the following: `A B C a b c 0 1 2 $ * + /` and the blank character.

ANS:

```
1  // Exercise 2.29 Solution: Display.java
2  // Program that prints a unicode character
3  // and its integer equivalent.
4
5  public class Display
6  {
7     public static void main( String args[] )
8     {
9        System.out.printf( "The character %c has the value %d\n",
10          'A', ( (int) 'A' ) );
11       System.out.printf( "The character %c has the value %d\n",
12          'B', ( (int) 'B' ) );
13       System.out.printf( "The character %c has the value %d\n",
14          'C', ( (int) 'C' ) );
15       System.out.printf( "The character %c has the value %d\n",
16          'a', ( (int) 'a' ) );
17       System.out.printf( "The character %c has the value %d\n",
18          'b', ( (int) 'b' ) );
19       System.out.printf( "The character %c has the value %d\n",
20          'c', ( (int) 'c' ) );
21       System.out.printf( "The character %c has the value %d\n",
22          '0', ( (int) '0' ) );
```

```
23         System.out.printf( "The character %c has the value %d\n",
24            '1', ( (int) '1' ) );
25         System.out.printf( "The character %c has the value %d\n",
26            '2', ( (int) '2' ) );
27         System.out.printf( "The character %c has the value %d\n",
28            '$', ( (int) '$' ) );
29         System.out.printf( "The character %c has the value %d\n",
30            '*', ( (int) '*' ) );
31         System.out.printf( "The character %c has the value %d\n",
32            '+', ( (int) '+' ) );
33         System.out.printf( "The character %c has the value %d\n",
34            '/', ( (int) '/' ) );
35         System.out.printf( "The character %c has the value %d\n",
36            ' ', ( (int) ' ' ) );
37      } // end main
38   } // end class Display
```

```
The character A has the value 65
The character B has the value 66
The character C has the value 67
The character a has the value 97
The character b has the value 98
The character c has the value 99
The character 0 has the value 48
The character 1 has the value 49
The character 2 has the value 50
The character $ has the value 36
The character * has the value 42
The character + has the value 43
The character / has the value 47
The character   has the value 32
```