2

Introduction to Java Applications

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.

Assignment Checklist

Name: _____ **Date:** _____

Section: _____

Exercises	Assigned: Circle assignments	Date Due
Prelab Activities		
Matching	YES NO	
Fill in the Blanks	YES NO	
Short Answer	YES NO	
Programming Output	YES NO	
Correct the Code	YES NO	
Lab Exercises		
Exercise 1 — Shapes	YES NO	
Follow-Up Question and Activity	1	
Exercise 2 — Number Calculations	YES NO	
Follow-Up Question and Activity	1	
Exercise 3 — Separating Digits	YES NO	
Follow-Up Questions and Activities	1, 2, 3	
Debugging	YES NO	
Postlab Activities		
Coding Exercises	1, 2, 3, 4, 5	
Programming Challenges	1, 2	

Prelab Activities

Matching

Name: _____ **Date:** _____

Section: _____

After reading Chapter 2 of *Java How to Program, 8/e*, answer the given questions. The questions are intended to test and reinforce your understanding of key concepts; you may answer the questions before or during the lab.

For each term in the left column, write the letter for the description from the right column that best matches the term.

Term	Description
F 1. `==`	a) Newline character.
J 2. `=`	b) Remainder operator.
H 3. `class`	c) A program that you execute with the `java` command.
E 4. `+`	d) Standard output object.
A 5. `\n`	e) Concatenation operator.
D 6. `System.out`	f) "Is equal to" operator
C 7. application	g) Where Java applications begin executing.
G 8. `main`	h) Introduces a class declaration.
B 9. `%`	i) Escape character.
I 10. `\`	j) Assignment operator.
N 11. `java`	k) Determines whether a statement (or set of statements) should execute.
M 12. `javac`	l) Used to specify classes required to compile a Java program.
P 13. `System.out.print` method	m) Compiles a Java program.
R 14. `System.out.println` method	n) Executes a Java application.
L 15. `import` declarations	o) Indicates that a method does not return any information when it completes its task.
Q 16. identifier	p) Displays information in the command window and does not position the output cursor to the beginning of the next line.
O 17. `void` keyword	q) A series of characters consisting of letters, digits, underscores and dollar signs that does not begin with a digit and does not contain spaces.
S 18. semicolon	r) Displays a line of information in the command window and automatically positions the output cursor to the beginning of the next line.
K 19. `if` statement	s) Ends every statement in a program.

Prelab Activities

Name:

Fill in the Blank

Name: _____ **Date:** _____

Section: _____

Fill in the blanks for each of the following statements:

20. By convention, all class names in Java begin with a(n) capital letter.

21. The empty string is a string that contains no characters.

22. Method `printf`'s first argument is a(n) format string that may consist of fixed text and format specifiers.

23. Every variable declared in a method must be initialized before it can be used in an expression.

24. `System.out` is known as the standard output object.

25. All variables must be declared with a(n) type and a(n) name before they can be used in a program.

26. End-of-line (single-line) comments begin with `//`.

27. `/*` begins a traditional (multiple-line) comment, and `*/` ends a traditional comment.

28. The `if` statement allows a program to make a decision based on the truth or falsity of some condition.

29. An `if` statement's condition is enclosed in parentheses.

Prelab Activities

Name:

Short Answer

Name: _____ **Date:** _____

Section: _____

Answer the following questions in the space provided. Your answers should be concise; aim for two or three sentences.

30. What does the `if` selection statement allow a program to do?

The `if` statement allows a program to make a decision based on the truth or falsity of some condition. If the condition is true, the statement in the `if` statement's body executes; otherwise, it is skipped.

31. What is a syntax error? Give an example.

A syntax error occurs when the compiler cannot recognize a statement. The compiler normally issues an error message to help the programmer identify and fix the incorrect statement. Syntax errors are violations of the language rules. For example, omitting the semicolon at the end of a statement is a syntax error.

32. What is the importance of a variable's name, type, size and value?

The name of a variable allows the programmer to access that variable to store a value or use the currently stored value. The type of the variable tells Java how to manipulate the variable properly and what operations are allowed to be performed on it. The size of a variable determines the range of values that the variable can represent. The value of a variable is the information that is currently stored.

33. What is an `import` declaration and where does it appear in a Java source code file?

The compiler uses `import` declarations to identify and load classes used in a Java program. When you use classes from the Java API, the compiler attempts to ensure that you use them correctly. The `import` declarations help the compiler locate the classes you intend to use. All `import` declarations in a Java file must appear before the class definition in that file.

34. Why do programmers insert comments in their code?

Programmers insert comments to document programs and improve program readability. Comments also help other people read and understand a program.

35. Why does a semicolon cause a logic error if placed immediately after the right parenthesis of an `if` statement?

Placing a semicolon after the parentheses that delimit the condition in an `if` statement causes a logic error because the body of the `if` statement becomes the empty statement, so the `if` statement itself does not perform an action, regardless of whether its condition is true or false. The intended body of the `if` statement will now become a statement (or statements) in sequence with the `if` statement and the body will always execute.

Prelab Activities

Name:

Programming Output

Name: _____ **Date:** _____

Section: _____

For each of the given program segments, read the code, and write the output in the space provided below each program. [*Note:* Do not execute these programs on a computer.]

36. What is the output of the following program?

```
1  public class Operator {
2
3     public static void main( String args[] )
4     {
5        int x = 30;
6        int y = 2;
7
8        System.out.println( x * y + 9 / 3 );
9     }
10 }
```

Your answer:

```
63
```

37. What is output by the following line of code?

```
1  System.out.println( ( 8 * 4 * 2 + 6 ) / 2 + 4 );
```

Your answer:

```
39
```

Prelab Activities

Name:

Programming Output

38. What is output by the following program for each of the input values 5, 7, 100, –7 and 0?

```
1  import java.util.Scanner;
2
3  public class Output
4  {
5     public static void main( String args[] )
6     {
7        int number;
8        Scanner input = new Scanner( System.in );
9
10       System.out.println( "Enter integer: " );
11       number = input.nextInt();
12
13       if ( number != 7 )
14          System.out.print( "Welcome " );
15
16       if ( ( number % 5 ) == 0 )
17          System.out.println( "To Java Programming" );
18    }
19 }
```

Your answer:

```
Enter integer: 5
Welcome To Java Programming
```

```
Enter integer: 7
```

```
Enter integer: 100
Welcome To Java Programming
```

```
Enter integer: -7
Welcome
```

```
Enter integer: 0
Welcome To Java Programming
```

Prelab Activities

Name:

Programming Output

39. What is output by the following program? Assume the user enters 12 for one execution of the program and 15 for a second execution.

```
1  import java.util.Scanner;
2
3  public class Compares
4  {
5     public static void main( String args[] )
6     {
7        int integer;
8        Scanner input = new Scanner( System.in );
9
10       System.out.println( "Enter an integer:" );
11       integer = input.nextInt();
12
13       if ( ( integer % 6 ) == 0 )
14          System.out.println( "Hello" );
15       else
16          System.out.println( "Good Bye" );
17    }
18 }
```

Your answer:

```
Enter an integer: 12
Hello
```

```
Enter an integer: 15
Good Bye
```

Prelab Activities

Name:

Programming Output

40. What is output by the following program?

```
1  public class Compares
2  {
3     public static void main( String args[] )
4     {
5        int x = 3;
6        int y = 9;
7        int z = 77;
8
9        if ( z == 77 )
10          System.out.print( "H" );
11
12       if ( z == 99 )
13          System.out.print( "M" );
14
15       if ( z < x )
16          System.out.print( "J" );
17
18       System.out.print( "E" );
19
20       if ( y == ( x * x ) )
21          System.out.print( "LL" );
22
23       System.out.print( "O" );
24
25       if ( x == y )
26          System.out.print( "W" );
27    }
28 }
```

Your answer:

```
HELLO
```

41. What is output by the program in Exercise 40 when `x = 11`, `y = 121` and `z = 10`?

Your answer:

```
JELLO
```

42. What is output by the program in Exercise 40 when `x = 5`, `y = 25` and `z = 99`?

Your answer:

```
MELLO
```

Prelab Activities

Name:

Programming Output

43. What is output by the program in Exercise 40 when x = 10, y = 9 and z = 8?

Your answer:

```
JEO
```

44. What is output by the program in Exercise 40 when x = 10, y = 10 and z = 99?

Your answer:

```
MEOW
```

Prelab Activities

Name:

Correct the Code

Name: _____ **Date:** _____

Section: _____

Determine if there is an error in each of the following program segments. If there is an error, specify whether it is a logic error or a compilation error, circle the error in the program and write the corrected code in the space provided after each problem. If the code does not contain an error, write "no error." [*Note:* There may be more than one error in each program segment.]

45. The following program should input the value of an integer into variable `num`:

```
1  import java.Scanner;
2
3  public class Output {
4
5     public static void main( String args[] )
6
7        int num
8        Scanner input = Scanner( in );
9
10       num = input.int();
11    }
12 }
```

Your answer:

```
1  import java.Scanner;
2
3  public class Output {
4
5     public static void main( String args[] )
6
7        int num;
8        Scanner input = new Scanner( System.in );
9
10       num = input.nextInt();
11    }
12 }
```

- On line 7, missing semicolon.
- On line 8, missing new keyword.
- On line 8, standard input object is System.in.
- `Scanner` class does not contain an `int` method. Use method `nextInt` on line 10 to retrieve an integer from the user.

Prelab Activities

Name:

Correct the Code

46. The following segment of code should declare an `int` variable `number` and assign the value of the expression `(5 + 3) * 2` to the variable:

```
1  int number;
2  number = 5 + 3 * 2;
```

Your answer:

```
1  int number;
2  number = ( 5 + 3 ) * 2;
```

- Missing parentheses on line 2.

47. The following code should determine whether variable `q` is equal to `100`:

```
1  int q = 100;
2
3  System.out.print( "q is" );
4
5  if ( q = 100 )
6     System.out.print( " equal to 100" );
7
8  if ( q ! 100 )
9     System.out.print( " not equal to 100" );
```

Your answer:

```
1  int q = 100;
2
3  System.out.print( "q is" );
4
5  if ( q == 100 )
6     System.out.print( " equal to 100" );
7
8  if ( q != 100 )
9     System.out.print( " not equal to 100" );
```

- Must use “is equal to” operator on line 5 and “is not equal to” operator on line 8.

48. The following code segment should determine whether an integer variable’s value is less than zero.

```
1  int x = 9;
2
3  if ( x < 0 );
4     System.out.println( "Variable x is less than zero" );
```

Prelab Activities

Name:

Correct the Code

Your answer:

```
1  int x = 9;
2
3  if ( x < 0 )
4     System.out.println( "Variable x is less than zero" );
```

- There should be no semicolon at the end of line 3.

49. The following program should output the integer value entered by the user:

```
1  import java.util.Scanner;
2
3  public class Display
4  {
5     public static void main( String args[] )
6     {
7        int num1;
8        Scanner input = new Scanner( System.in );
9
10       System.out.println( "Enter first integer:" );
11
12       Scanner.nextInt( num1 );
13       System.out.println( num1 );
14    }
15 }
```

Your answer:

```
1  import java.util.Scanner;
2
3  public class Display
4  {
5     public static void main( String args[] )
6     {
7        int num1;
8        Scanner input = new Scanner( System.in );
9
10       System.out.println( "Enter first integer:" );
11
12       num1 = input.nextInt();
13       System.out.println( num1 );
14    }
15 }
```

- The call on line 12 must use the `input` variable we created on line 8, not the class name `Scanner`.
- The `nextInt` method takes no arguments. Its return value should be assigned to variable `num1`.

Prelab Activities

Name:

Correct the Code

50. The following code should compare two integers to determine if they are not equal.

```
1  int x = 9;
2  int y = 3;
3
4  if ( x =! y )
5     System.out.println( "Variable x and y are not equal" );
```

Your answer:

```
1  int x = 9;
2  int y = 3;
3
4  if ( x != y )
5     System.out.println( "Variable x and y are not equal" );
```

- On line 4, the "not equals to" operator should be written as `!=`.

Lab Exercises

Lab Exercise 1 — Shapes

Name: _____ **Date:** _____

Section: _____

This problem is intended to be solved in a closed-lab session with a teaching assistant or instructor present. The problem is divided into six parts:

1. Lab Objectives
2. Description of the Problem
3. Sample Output
4. Program Template (Fig. L 2.1)
5. Problem-Solving Tips
6. Follow-Up Question and Activity

The program template represents a complete working Java program, with one or more key lines of code replaced with comments. Read the problem description and examine the sample output; then study the template code. Using the problem-solving tips as a guide, replace the `/* */` comments with Java code. Compile and execute the program. Compare your output with the sample output provided. Then answer the follow-up question. The source code for the template is available at `www.pearsonhighered.com/deitel`.

Lab Objectives

This lab was designed to reinforce programming concepts from Chapter 2 of *Java How to Program: 8/e*. In this lab, you will practice:

- Using `System.out.println` to output text and characters to the command window.
- Compiling and executing Java applications.

The follow-up question and activity also will give you practice:

- Modifying an existing program to perform a different task.

Description of the Problem

Write an application that displays the shapes shown in the sample output using asterisks.

Sample Output

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

Lab Exercises

Name:

Lab Exercise 1 — Shapes

Program Template

```
1  // Lab 1: Shapes.java
```

Fig. L 2.1 | Shapes.java.

```
2  // Program draws four shapes to the command window.
3
4  public class Shapes
5  {
6     public static void main( String args[] )
7     {
8        /* write a series of statements that will print the shapes
9           to the command window */
10    } // end main
11 } // end class Shapes
```

Solution

```
1  // Lab 1: Shapes.java
2  // Program draws four shapes to the command window.
3
4  public class Shapes
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       System.out.println( "*********     ***       *        *    ");
17    } // end main
18 } // end class Shapes
```

Problem-Solving Tips

1. Notice that there are nine rows of asterisks. Write nine `System.out.println` statements.
2. Compile and execute your program in a command window. Change to the directory where the program is stored and type `javac Shapes.java` to compile your program; then to execute it type `java Shapes` to launch the JVM.
3. Be sure to follow the spacing and indentation conventions discussed in *Java How to Program: 8/e*.
4. If you have any questions as you proceed, ask your lab instructor for assistance.

Follow-Up Question and Activity

1. Modify the program so that it includes a triangle in its output. The triangle should have a base containing 17 asterisks.

Lab Exercises

Name:

Lab Exercise 1 — Shapes

Sample Output

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

Solution

```
1  // Lab 1: Shapes.java
2  // Program draws four shapes to the command window.
3
4  public class Shapes
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       System.out.println( "*********     ***       *        *      *****************");
17    } // end main
18 } // end class Shapes
```

Lab Exercises

Name:

Lab Exercise 2 — Number Calculations

Name: _____ **Date:** _____

Section: _____

This problem is intended to be solved in a closed-lab session with a teaching assistant or instructor present. The problem is divided into six parts:

1. Lab Objectives
2. Description of the Problem
3. Sample Output
4. Program Template (Fig. L 2.2)
5. Problem-Solving Tips
6. Follow-Up Question and Activity

The program template represents a complete working Java program, with one or more key lines of code replaced with comments. Read the problem description and examine the sample output; then study the template code. Using the problem-solving tips as a guide, replace the `/* */` comments with Java code. Compile and execute the program. Compare your output with the sample output provided. Then answer the follow-up question. The source code for the template is available at `www.pearsonhighered.com/deitel`.

Lab Objectives

This lab was designed to reinforce programming concepts from Chapter 2 of *Java How to Program: 8/e*. In this lab, you will practice:

- Using the `Scanner` class to obtain input from the user.
- Using `printf` to output information to the user.
- Using arithmetic operators to perform calculations.
- Using `if` statements to make decisions based on the truth or falsity of a condition.
- Using relational operators to compare variable values.

The follow-up question and activity will give you practice:

- Understanding a common programming error with `if` statements

Description of the Problem

Write an application that inputs three integers from the user and displays the sum, average, product, smallest and largest of the numbers. [*Note:* The calculation of the average in this exercise should result in an integer representation of the average. So if the sum of the values is 7, the average should be 2, not 2.3333….]

Lab Exercises

Name:

Lab Exercise 2 — Number Calculations

Sample Output

```
Enter first integer: 10
Enter second integer: 20
Enter third integer: 30

For the numbers 10, 20 and 30
Largest is 30
Smallest is 10
Sum is 60
Product is 6000
Average is 20
```

Program Template

```
1  // Lab 2: Calculate2.java
2  // Performing calculations.
3  import java.util.Scanner;
4
5  public class Calculate2
6  {
7     public static void main( String args[] )
8     {
9        Scanner input = new Scanner( System.in );
10
11       int number1; // first number
12       int number2; // second number
13       int number3; // third number
14       int largest; // largest value
15       int smallest; // smallest value
16       int sum; // sum of numbers
17       int product; // product of numbers
18       int average; // average of numbers
19
20       /* write a series of statements to read in three numbers and assign them
21          to number1, number2, and number3 */
22
23       largest = number1; // assume number1 is the largest
24       smallest = number1; // assume number1 is the smallest
25
26       /* write code here that compares all three integers and sets the
27          largest and smallest accordingly */
28
29       // perform calculations
30       sum = number1 + number2 + number3;
31       /* write statements to calculate the product and the average */
32
33       /* write statements that display the results */
34    } // end main
35 } // end class Calculate2
```

Fig. L 2.2 | Calculate2.java.

Lab Exercises

Name:

Lab Exercise 2 — Number Calculations

Problem-Solving Tips

1. Prompt the user for three integer values and use `Scanner` method `nextInt` to read them into their respective `int` variables.
2. Use a series of `if` statements to determine the smallest and largest numbers. You must use relational operators in the `if` conditions to compare two numbers at a time.
3. Calculate the sum, product and average, and assign them to variables called `sum`, `product` and `average`, respectively. Then, display the results in an information message dialog.
4. Test your program thoroughly using different test inputs and determine whether your program produces the correct results. Try entering 10, 20, and 30 and see if your results match the sample output above.
5. Be sure to follow the spacing and indentation conventions discussed in *Java How to Program: 8/e*.
6. If you have any questions as you proceed, ask your lab instructor for assistance.

Solution

```
1  // Lab 2: Calculate2.java
2  // Performing calculations.
3  import java.util.Scanner;
4
5  public class Calculate2
6  {
7     public static void main( String args[] )
8     {
9        Scanner input = new Scanner( System.in );
10
11       int number1; // first number
12       int number2; // second number
13       int number3; // third number
14       int largest; // largest value
15       int smallest; // smallest value
16       int sum; // sum of numbers
17       int product; // product of numbers
18       int average; // average of numbers
19
20       System.out.print( "Enter first integer: " ); // prompt for input
21       number1 = input.nextInt(); // read first number
22       System.out.print( "Enter second integer: " ); // prompt for input
23       number2 = input.nextInt(); // read second number
24       System.out.print( "Enter third integer: " ); // prompt for input
25       number3 = input.nextInt(); // read third number
26
27       // determine largest value
28       largest = number1; // assume number1 is the largest
29
30       if ( number2 > largest ) // determine whether number2 is larger
31          largest = number2;
32
33       if ( number3 > largest ) // determine whether number3 is larger
34          largest = number3;
35
36       // determine smallest value
37       smallest = number1; // assume number1 is the smallest
38
```

Lab Exercises

Name:

Lab Exercise 2 — Number Calculations

```
39         if ( number2 < smallest ) // determine whether number2 is smallest
40            smallest = number2;
41
42         if ( number3 < smallest ) // determine whether number3 is smallest
43            smallest = number3;
44
45         // perform calculations
46         sum = number1 + number2 + number3;
47         product = number1 * number2 * number3;
48         average = sum / 3;
49
50         // print results
51         System.out.printf( "\nFor the numbers %d, %d and %d\n",
52            number1, number2, number3 );
53         System.out.printf( "Largest is %d\n", largest );
54         System.out.printf( "Smallest is %d\n", smallest );
55         System.out.printf( "Sum is %d\n", sum);
56         System.out.printf( "Product is %d\n", product );
57         System.out.printf( "Average is %d\n", average );
58      } // end main
59   } // end class Calculate2
```

Follow-Up Question and Activity

1. Place a semicolon at the end of the condition of an `if` statement in your solution that is used to help determine the largest and smallest values. What happens? Explain.

   A logic error occurs. The semicolon will cause the body of the `if` statement to be empty, so the `if` statement itself will perform no action, regardless of whether its condition is true. Worse yet, the intended body statement of the `if` statement will now become a statement in sequence with the `if` statement and will always execute. The body statement would then always replace the largest or smallest values with a value that may not be larger or smaller. If no later statement replaces the incorrect value, then the wrong value will be displayed.

Lab Exercises

Name:

Lab Exercise 3 — Separating Digits

Name: _____ **Date:** _____

Section: _____

This problem is intended to be done in a closed-lab session with a teaching assistant or instructor present. The problem is divided into six parts:

1. Lab Objectives
2. Description of the Problem
3. Sample Output
4. Program Template (Fig. L 2.3)
5. Problem-Solving Tips
6. Follow-Up Questions and Activities

The program template represents a complete working Java program, with one or more key lines of code replaced with comments. Read the problem description and examine the sample output; then study the template code. Using the problem-solving tips as a guide, replace the `/* */` comments with Java code. Compile and execute the program. Compare your output with the sample output provided. Then answer the follow-up questions. The source code for the template is available at `www.pearsonhighered.com/deitel`.

Lab Objectives

This lab was designed to reinforce programming concepts from Chapter 2 of *Java How to Program: 8/e*. In this lab you will practice:

- Using the remainder operator (`%`) to determine the remainder of a division operation.
- Demonstrating that integer division yields integer results.

The follow-up questions and activities also will give you practice:

- Examining what happens during program execution when the user enters invalid input.
- Using type `double` to declare floating-point variables.
- Adapting a program to solve a similar problem.

Description of the Problem

Write an application that inputs one number consisting of five digits from the user, separates the number into its individual digits and prints the digits separated from one another by three spaces each. For example, if the user types in the number 42339, the program should print "`4  2  3  3  9`." Assume that the user enters the correct number of digits.

Sample Output

```
Enter five digit integer: 12345
Digits in 12345 are 1  2  3  4  5
```

Lab Exercises

Name:

Lab Exercise 3 — Separating Digits

Program Template

```
1  // Lab 3: Five.java
2  // Separating the digits in a five-digit number.
3  import java.util.Scanner;
4
5  public class Five
6  {
7     public static void main( String args[] )
8     {
9        Scanner input = new Scanner( System.in );
10
11       int number; // number input by user
12       int digit1; // first digit
13       int digit2; // second digit
14       int digit3; // third digit
15       int digit4; // fourth digit
16       int digit5; // fifth digit
17
18       System.out.print( "Enter five digit integer: " ); // prompt
19       number = input.nextInt(); // read number
20
21       // determine the 5 digits
22       digit1 = number / 10000;
23       digit5 = number % 10;
24       /* write code here that will separate the remainder of the digits in the
25          variable "number" and assign each one to the corresponding integer
26          variable */
27
28       /* write a statement that displays each digit separated by three spaces. */
29    } // end main
30 } // end class Five
```

Fig. L 2.3 | Five.java.

Problem-Solving Tips

1. The input data consists of one integer, so you will use an `int` variable to represent it. Note that the description indicates that one five-digit number is to be input—not five separate digits.
2. You will use a series of statements to "break down" the number into its individual digits, using integer arithmetic with remainder (`%`) and division (`/`) calculations.
3. After the number has been input, divide the number by `10000` to get the first digit. Why does this operation work? In Java, dividing an integer by an integer yields an integer result. Because the number input is five digits long, dividing it by `10000` gives the leftmost digit. For example, `42339 / 10000` evaluates to `4` because `10000` divides into `42339` four times. The remainder is truncated in integer arithmetic.
4. Change the number to a four-digit number, using the remainder operator to obtain the remainder after the number is divided by `10000`—in this case, the rightmost four digits. For example, `42339 % 10000` results in `2339`.
5. Repeat this pattern of division and remainder calculations. Each time, the number used in the division and remainder calculations is reduced by a factor of `10`. The first digit is obtained by dividing the five-digit number by `10000`. Then, the variable containing the number is assigned the remainder after the five-digit number is divided by `10000`. After the number is changed to a four-digit number, perform division and remainder calculations with `1000`; after the number is changed to a three-digit number, perform division and remainder calculations with `100`, and so on.

Lab Exercises

Name:

Lab Exercise 3 — Separating Digits

6. Be sure to follow the spacing and indentation conventions discussed in *Java How to Program: 8/e*.
7. If you have any questions as you proceed, ask your lab instructor for assistance.

Solution

```
1  // Lab 3: Five.java
2  // Separating the digits in a five-digit number.
3  import java.util.Scanner;
4
5  public class Five
6  {
7     public static void main( String args[] )
8     {
9        Scanner input = new Scanner( System.in );
10
11       int number; // number input by user
12       int digit1; // first digit
13       int digit2; // second digit
14       int digit3; // third digit
15       int digit4; // fourth digit
16       int digit5; // fifth digit
17
18       System.out.print( "Enter five digit integer: " ); // prompt
19       number = input.nextInt(); // read number
20
21       // determine the 5 digits
22       digit1 = number / 10000;
23       digit2 = number % 10000 / 1000;
24       digit3 = number % 1000 / 100;
25       digit4 = number % 100 / 10;
26       digit5 = number % 10;
27
28       // output results
29       System.out.printf( "Digits in %d are %d  %d  %d  %d  %d\n",
30          number, digit1, digit2, digit3, digit4, digit5 );
31    } // end main
32 } // end class Five
```

Follow-Up Questions and Activities

1. What are the results of the following expressions?

   24 / 5 = ___4___

   18 % 3 = ___0___

   13 % 9 = ___4___

   13 / 2 % 2 = ___0___

Lab Exercises

Name:

Lab Exercise 3 — Separating Digits

2. What happens when the user inputs a number that is less than five digits long? Why? What is the output when the user enters 1763?
   ANS: If a user enters a number that is less than five digits long, leading zeros are added. With a number with less than five digits, line 22 will set `digit1` equal to 0. If the user enters 1763, the program outputs `0 1 7 6 3`.
3. The program you completed in this lab exercise reads from the user a number with multiple digits and separates the digits. Write a program that inputs the individual digits that compose a larger number. Then use multiplication and addition operations to "assemble" the larger number.

Solution

```
1  // BuildFive.java
2  // Program builds a five-digit number from five single digits
3  import java.util.Scanner;
4
5  public class BuildFive
6  {
7     public static void main( String args[] )
8     {
9        Scanner input = new Scanner( System.in );
10
11       int number;
12       int digit1; // first digit of number
13       int digit2; // second digit of number
14       int digit3; // third digit of number
15       int digit4; // fourth digit of number
16       int digit5; // fifth digit of number
17
18       // read the five digits from user and convert them to integers
19       System.out.print( "Enter first digit: " ); // prompt
20       digit1 = input.nextInt(); // read number
21
22       System.out.print( "Enter second digit: " ); // prompt
23       digit2 = input.nextInt(); // read number
24
25       System.out.print( "Enter third digit: " ); // prompt
26       digit3 = input.nextInt(); // read number
27
28       System.out.print( "Enter fourth digit: " ); // prompt
29       digit4 = input.nextInt(); // read number
30
31       System.out.print( "Enter fifth digit: " ); // prompt
32       digit5 = input.nextInt(); // read number
33
34       // compose the five-digit integer
35       number = digit1 * 10000;
36       number = number + digit2 * 1000;
37       number = number + digit3 * 100;
38       number = number + digit4 * 10;
39       number = number + digit5 * 1;
40
41       // create the result string
42       System.out.printf( "Integer composed from %d  %d  %d  %d  %d is %d.\n",
43          digit1, digit2, digit3, digit4, digit5, number );
44    } // end main
45 } // end class BuildFive
```

Lab Exercises

Name:

Lab Exercise 3 — Separating Digits

```
Enter first digit: 1
Enter second digit: 2
Enter third digit: 3
Enter fourth digit: 4
Enter fifth digit: 5
Integer composed from 1  2  3  4  5 is 12345.
```

Lab Exercises

Name:

Debugging

Name: _____ **Date:** _____

Section: _____

The program in this section does not compile. Fix all the compilation errors so that the program will compile successfully. Once the program compiles, execute the program, and compare its output with the sample output; then eliminate any logic errors that may exist. The sample output demonstrates what the program's output should be once the program's code is corrected. The source code is available at the Web sites `www.pearsonhighered.com/deitel`.

Sample Output

```
Enter first integer:
5
Enter second integer:
3
Enter third integer:
2
The sum is 10
The product is 30
The average is 3
```

Broken Code

```
1  /* Chapter 2 of Java How to Program: 8/e
2     Debugging Problem /
3
4  public class Arithmetic
5  {
6  import java.util.Scanner;
7
8     public static void main( String args[] )
9     {
10       Scanner input = new Scanner( System.in );
11       int num2
12       int num3
13       int sum
14       int product
15       int average
16
17       System.out.println( "Enter first integer:" );
18       num1 == input.nextInt();
19
20       System.out.println( "Enter second integer:" );
21       num2 == input.nextInt();
22
23       System.out.println( "Enter third integer: );
24       num3 == input.nextInt();
25
```

Fig. L 2.4 | `Arithmetic.java` (Part 1 of 2.)

Lab Exercises

Name:

Debugging

```
26         sum = num1 + num2 + num3;
27         product = num1 * num2 * num3;
28         average = ( num1 + num2 + num3 ) / 3;
29
30         System.out.printf( "The sum is %d\nThe product is %d\nThe average is %d\n", sum,
31            product, average );
32      }
33   } // end class Arithmetic
```

Fig. L 2.4 | `Arithmetic.java`. (Part 2 of 2.)

Solution

```
1  /* Chapter 2 of Java How to Program: 8/e
2     Debugging Problem */
3  import java.util.Scanner;
4
5  public class Arithmetic
6  {
7     public static void main( String args[] )
8     {
9        Scanner input = new Scanner( System.in );
10       int num1;
11       int num2;
12       int num3;
13       int sum;
14       int product;
15       int average;
16
17       System.out.println( "Enter first integer:" );
18       num1 = input.nextInt();
19
20       System.out.println( "Enter second integer:" );
21       num2 = input.nextInt();
22
23       System.out.println( "Enter third integer:" );
24       num3 = input.nextInt();
25
26       sum = num1 + num2 + num3;
27       product = num1 * num2 * num3;
28       average = ( num1 + num2 + num3 ) / 3;
29
30       System.out.printf( "The sum is %d\nThe product is %d\nThe average is %d\n", sum,
31          product, average );
32    }
33 } // end class Arithmetic
```

List of Errors

- Forgetting or mistyping one of the delimiters of a multiple line comment (as in the closing delimiter on line 2) is a compilation error.
- There are no semicolons after the declarations on lines 11–15. Compilation error.
- `import` declaration appears in class declaration. It should be placed at line 3. Compilation error.
- Lines 18, 21, 24 attempt to use the equality operator to assign values to variables `num1`, `num2`, and `num3`.

Lab Exercises

Name:

Debugging

- Variable `num1` is not declared. It must be declared before line 18. Compilation error.
- Forgetting the closing double quotes on a `String` (line 23) is a compilation error.

Postlab Activities

Coding Exercises

Name: _____ **Date:** _____

Section: _____

These coding exercises reinforce the lessons learned in the lab and provide additional programming experience outside the classroom and laboratory environment. They serve as a review after you have successfully completed the *Prelab Activities* and *Lab Exercises*.

For each of the following problems, write a program or a program segment that performs the specified action:

1. Write an `import` declaration which indicates that the program uses the `Scanner` class.

```
import java.util.Scanner;
```

2. Write a statement that declares a `Scanner` variable and assigns it a `Scanner` object that reads from `System.in`.

```
Scanner input = new Scanner( System.in );
```

3. Write a line of code that prompts the user to input an integer.

```
System.out.println( "Please enter an integer:" );
```

4. Write a line of code that uses the `Scanner` object created in *Coding Exercise 2* to read an integer and assign it to variable `number`.

```
int number = input.nextInt();
```

5. Write code that squares the integer variable from *Coding Exercise 4*, stores the new value in `int` variable `square` and displays the resulting value using `System.out.printf`.

```
1 int square = number * number;
2 System.out.printf( "%d\n", square );
```

Postlab Activities

Name:

Programming Challenges

Name: _____ **Date:** _____

Section: _____

The *Programming Challenges* are more involved than the *Coding Exercises* and may require a significant amount of time to complete. Write a Java program for each of the problems in this section. The answers to these problems are available at `www.pearsonhighered.com/deitel`. Pseudocode, hints or sample outputs are provided for each problem to aid you in your programming.

1. 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 = $2r$

   circumference = $2\pi r$

   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.

   Hints:

   - In Chapter 2 of *Java How to Program: 8/e*, we do not introduce a format specifier that is capable of printing floating-point numbers. For this exercise, output all floating-point values using the `%f` format specifier. For example if you want to output the computed area, you would write:

     ```
     System.out.printf( "Area is %f\n", ( Math.PI * radius * radius ) );
     ```

   - Use (`\n`) to force a new line.
   - Use end-of-line comments (`//`) to clarify difficult concepts in the program.
   - Your output should appear as follows:

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

Postlab Activities

Name:

Programming Challenges

Solution

```
1  // Programming Challenge 1
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
```

Postlab Activities

Name:

Programming Challenges

2. 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.]

Hints:

- This program requires one input from the user and an `if` statement that tests whether the integer is divisible by 2 using the remainder operator.
- Your output should appear as follows:

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

```
Enter integer: 14
Number is even
```

Solution

```
1  // Programming Challenge 2
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
```