

Revision Exercise 3

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

- a. The **default** case is required in the **switch** selection structure.
- b. The **break** statement is required in the default case of a **switch** selection structure.
- c. The expression (**x > y && a < b**) is true if either **x > y** is true or **a < b** is true.
- d. An expression containing the **||** operator is true if either or both of its operands is true.

(2) Write a Java statement or a set of Java statements to accomplish each of the following:

- a. Sum the odd integers between 1 and 99 using a **for** structure. Assume the integer variables **sum** and **count** have been declared.
- b. Calculate the value of **2.5** raised to the power of **3** using the **pow** method.
- c. Print the integers from 1 to 20 using a **while** loop and the counter variable **x**. Assume that the variable **x** has been declared but not initialized. Print only five integers per line. [*Hint*: Use the calculation **x % 5**. When the value of this is 0, print a newline character; otherwise, print a tab character. Assume this is an application—use the **System.out.println()** method to output the newline character and use the **System.out.print('\t')** method to output the tab character.]
- d. Repeat Exercise 2 c) using a **for** structure.

(3) Find the error in each of the following code segments and explain how to correct it.

a. **x = 1;**

```
while ( x <= 10 );  
    x++;  
}
```

b. **for (y = .1; y != 1.0; y += .1)**
 System.out.println(y);

c. **switch (n) {**
 case 1:

```

        System.out.println( "The number is 1" );
    case 2:
        System.out.println( "The number is 2" );
        break;
    default:
        System.out.println( "The number is not 1 or 2" );
        break;
}

```

- d. The following code should print the values 1 to 10.

```

n = 1;

while ( n < 10 )
    System.out.println( n++ );

```

(4)

Find the error in each of the following. [*Note:* There may be more than one error.]

- a.

```
For ( x = 100, x >= 1, x++ )
    System.out.println( x );
```
- b. The following code should print whether integer **value** is odd or even:

```
switch ( value % 2 ) {
    case 0:
        System.out.println( "Even integer" );
    case 1:
        System.out.println( "Odd integer" );
}
```
- c. The following code should output the odd integers from 19 to 1:

```
for ( x = 19; x >= 1; x += 2 )
    System.out.println( x );
```
- d. The following code should output the even integers from 2 to 100:

```
counter = 2;
do {
    System.out.println( counter );
    counter += 2;
} While ( counter < 100 );
```

- (5) Write an application that finds the smallest of several integers. Assume that the first value read specifies the number of values to input from the user.

(6) The *factorial* method is used frequently in probability problems. The factorial of a positive integer n (written $n!$ and pronounced "n factorial") is equal to the product of the positive integers from 1 to n . Write an application that evaluates the factorials of the integers from 1 to 5. Display the results in tabular format. What difficulty might prevent you from calculating the factorial of 20?

(7) Assume $i = 1$, $j = 2$, $k = 3$ and $m = 2$. What does each of the following statements print? Are the parentheses necessary in each case?

- a. `System.out.println(i == 1);`
- b. `System.out.println(j == 3);`
- c. `System.out.println(i >= 1 && j < 4);`
- d. `System.out.println(m <= 99 & k < m);`
- e. `System.out.println(j >= i || k == m);`
- f. `System.out.println(k + m < j | 3 - j >= k);`
- g. `System.out.println(!(k > m));`

(8) What does the following program segment do?

```
for ( i = 1; i <= 5; i++ ) {
    for ( j = 1; j <= 3; j++ ) {
        for ( k = 1; k <= 4; k++ )
            System.out.print( '*' );
        System.out.println();
    }
    System.out.println();
}
```

(9) Calculate the value of π from the infinite series

$$\pi = 4 - \frac{4}{3} + \frac{4}{5} - \frac{4}{7} + \frac{4}{9} - \frac{4}{11} + \cdots$$

Print a table that shows the value of π approximated by one term of this series, by two terms, by three terms, etc. How many terms of this series do you have to use before you first get 3.14? 3.141? 3.1415? 3.14159?

(10) (*De Morgan's Laws*) In this chapter, we discussed the logical operators `&&`, `&`, `||`, `|`, `^` and `!`. De Morgan's Laws can sometimes make it more convenient for us to express a logical expression. These laws state that the expression `!(condition1 && condition2)` is logically equivalent to the expression `(!condition1 || !condition2)`. Also, the expression `!(condition1 || condition2)` is logically equivalent to the expression `(!condition1 && !condition2)`. Use De Morgan's Laws to write equivalent expressions for each of the following, and then write a program to show that both the original expression and the new expression in each case are equivalent:

- a. `!(x < 5) && !(y >= 7)`
- b. `!(a == b) || !(g != 5)`
- c. `!((x <= 8) && (y > 4))`
- d. `!((i > 4) || (j <= 6))`

Answers:

(1)

- a. False. The **default** case is optional. If no default action is needed, then there is no need for a **default** case.
- b. False. The **break** statement is used to exit the **switch** structure. The **break** statement is not required for the last case in a **switch** structure.
- c. False. Both of the relational expressions must be true in order for the entire expression to be true when using the `&&` operator.
- d. True.

(2)

- a. `sum = 0;`
`for (count = 1; count <= 99; count += 2)`
`sum += count;`
- b. `Math.pow(2.5, 3)`
- c. `x = 1;`

```
while ( x <= 20 ) {
    System.out.print( x );

    if ( x % 5 == 0 )
        System.out.println();
    else
```

```

        System.out.print( '\t' );

        ++x;
    }
d. for ( x = 1; x <= 20; x++ ) {

    System.out.print( x );

    if ( x % 5 == 0 )
        System.out.println();
    else
        System.out.print( '\t' );
}

```

or

```

for ( x = 1; x <= 20; x++ )

    if ( x % 5 == 0 )
        System.out.println( x );
    else
        System.out.print( x + "\t" );

```

(3)

- a. Error: The semicolon after the **while** header causes an infinite loop and there is a missing left brace.

Correction: Replace the semicolon by a { or remove both the ; and the }.

- b. Error: Using a floating-point number to control a **for** repetition structure may not work because floating-point numbers are represented approximately by most computers.

Correction: Use an integer, and perform the proper calculation in order to get the values you desire.

```

for ( y = 1; y != 10; y++ )
    System.out.println( (float) y / 10 );

```

- c. Error: Missing **break** statement in the statements for the first **case**.

Correction: Add a **break** statement at the end of the statements for the first **case**. Note that this is not necessarily an error if the programmer wants the statement of **case 2:** to execute every time the **case 1:** statement executes.

- d. Error: Improper relational operator used in the **while** repetition-continuation condition.

Correction: Use **<=** rather than **<** or change **10** to **11**.

(4)

- a. **ANS:** The **F** in **for** should be lowercase. Semicolons should be used in the **for** header instead of commas. **++** should be **--**.
- b. **ANS:** A **break** statement should be placed in **case 0**.
- c. **ANS:** **+=** should be **-=**.
- d. **ANS:** The **W** in **While** should be lowercase. **<** should be **<=**.

(5)

```
// Small.java
// Program finds the smallest of several letters
import javabook.*;
public class Small {
    public static void main( String args[] )
    {
        MainWindow mainWindow = new MainWindow("Find Smallest letter
program");
        InputBox  inputBox      = new InputBox(mainWindow);
        OutputBox outputBox     = new OutputBox(mainWindow);
        mainWindow.show();
        outputBox.show();
        int smallest = 0, temp = 0, number;
        number = inputBox.getInteger( "Enter number of integers:" );
        if ( number == 0 )
            System.exit( 0 );
        for ( int x = 1; x <= number; x++ ) {
            temp = inputBox.getInteger( "Enter integer:" );
            if ( x == 1 )
                smallest = temp;
            else if ( temp < smallest )
                smallest = temp;
        }
        outputBox.println("Smallest Integer is: " + smallest);
    }
}
```

(6)

```
// Factorial.java
// Program calculates factorials
public class Factorial {
    public static void main( String args[] )
    {
        int fact;
        String output = "X\tX!\n";
        for ( int z = 1; z <= 5; z++ ) {
            fact = 1;
            for ( int w = 1; w <= z; w++ )
                fact *= w;
            output += "\n" + z + "\t" + fact;
        }
        System.out.println(output);
    }
}
```

(7)

- a. **ANS:** True.
- b. **ANS:** False.
- c. **ANS:** True.
- d. **ANS:** False.
- e. **ANS:** True.
- f. **ANS:** False.
- g. **ANS:** False.

(8)

(9)

// Pi.java

// Program calculates Pi

```
public class Pi {
    public static void main( String args[] )
    {
        double piValue = 0, num = 4.0, denom = 1.0;
        int accuracy = 400000;
        String output = "Accuracy: " + accuracy;
        output += "\nTerm\t\tPi\n";
        System.out.println("Please wait.....");
        for ( int term = 1; term <= accuracy; term++ ) {
            if ( term % 2 != 0 )
                piValue += num / denom;
            else
                piValue -= num / denom;
            output += "\n" + term + "\t\t" + piValue;
            denom += 2.0;
        }
        System.out.println(output);
    }
}
```

(10)

// DeMorgan.java

// Program tests DeMorgan's laws

```
public class DeMorgan {
    public static void main( String args[] )
    {
        int x = 6, y = 0;
        String result = "";
```



```

// part a
if ( !( x < 5 ) && !( y >= 7 ) )
    result += "\n!( x < 5 ) && !( y >= 7 )";
if ( !( ( x < 5 ) || ( y >= 7 ) ) )
    result += "\n!( ( x < 5 ) || ( y >= 7 )";
int a = 8, b = 22, g = 88;
// part b
if ( !( a == b ) || !( g != 5 ) )
    result += "\n!( a == b ) || !( g != 5 )";
if ( !( ( a == b ) && ( g != 5 ) ) )
    result += "\n!( ( a == b ) && ( g != 5 ) )";
x = 8;
y = 2;
// part c
if ( !( ( x <= 8 ) && ( y > 4 ) ) )
    result += "\n!( ( x <= 8 ) && ( y > 4 ) )";
if ( !( x <= 8 ) || !( y > 4 ) )
    result += "\n!( x <= 8 ) || !( y > 4 )";
int i = 0, j = 7;
// part d
if ( !( ( i > 4 ) || ( j <= 6 ) ) )
    result += "\n!( ( i > 4 ) || ( j <= 6 ) )";
if ( !( i > 4 ) && !( j <= 6 ) )
    result += "\n!( i > 4 ) && !( j <= 6 )";
System.out.println(result);
}
}

```