**Chapter 3 - Selections**

|  |  |  |  |
| --- | --- | --- | --- |
| Sections | Pages | Review Questions | Programming Exercises |
| 3.1-3.8 | 68-94 | 1-37 | 2,4,6,8,10,12,14,18,20 |

**Boolean Datatype**

1. Boolean Variables

A *boolean* (*logical*) variable takes one of these values: *true* or *false*. For instance consider this *snippet* of code:

boolean isTall;  
 isTall = true;  
 System.out.println( isTall );

which displays: *true*

1. Boolean Expression. For example, suppose that we have two integers *x* and *y*. Then, *x* > *y* is a boolean expression. That is, it evaluates to *true* or *false*. For example:

Scanner input = new Scanner( System.in );  
 System.out.print( "What is your height? " );  
 int height = input.nextInt();   
   
 boolean isTall;  
   
 isTall = height > 72;

**IF Statement**

1. An *If* statement is used to implement optional behavior in a program. In *If* statement evaluates a boolean expression. If the expression is *true*, the some additional code is executed. If the expression is *false*, the additional code is skipped. Suppose that we want to read in someone’s height and display their height. In addition, if the person is over 72 inches we want to display a message that says the person is tall.

Scanner input = new Scanner( System.in );  
 System.out.print( "What is your height? " );  
 int height = input.nextInt();   
   
 System.out.println( "You are " + height + " inches tall" );  
   
 if( height > 72 )  
 {  
 System.out.println( "You are really tall" );  
 }

1. Notice that the code above did not use a boolean variable for any reason. We could re-write the code above to utilize a boolean variable.

Scanner input = new Scanner( System.in );  
 System.out.print( "What is your height? " );  
 int height = input.nextInt();   
   
 boolean isTall;  
   
 isTall = height > 72;  
   
 System.out.println( "You are " + height + " inches tall" );  
   
 if( isTall )  
 {  
 System.out.println( "You are really tall" );  
 }

In this simple case, it is probably preferable to use the first set of code, without the boolean variable. In real systems, there might be many places where we need to check the same condition (*e.g.* is height greater than 72 inches). In such cases, it would be better to use a boolean variable.

|  |  |  |
| --- | --- | --- |
| **Not Preferred** |  | **Preferred** |
|  |  |  |
| if( height > 72 ) {  // Do something }  ...  if( height > 72 ) {  // Do something }  ...  if( height > 72 ) {  // Do something }  ... |  | boolean isTall = height > 72;  if( isTall ) {  // Do something }  ...  if( isTall ) {  // Do something }  ...  if( isTall ) {  // Do something }  ... |

Can you think of at least two reasons why the code on the right is preferred?

**IF/ELSE Statement**

An *If/Else* statement is used to implement alternative actions, depending on whether the boolean expression is *true* or *false.* some optional behavior in a program. Suppose, in the preceding example that we want to print “not tall” if a person is 72 inches or smaller and “tall” if the person exceeds 72 inches.

if( height <= 72 )  
 {  
 System.out.println( "You are not tall" );  
 }  
 else  
 {  
 System.out.println( "You are tall" );  
 }

**Nested IF Statement**

1. We can nest IF statements inside of IF (or IF/ELSE) statements when we have more than two alternatives. Suppose we want to display the following message, depending on the person’s height.

|  |  |
| --- | --- |
| **Height** | **Message** |
|  | Small |
|  | Medium |
|  | Tall |

1. Code

if( height < 62 )  
 {  
 System.out.println( "You are not short" );  
 }  
 else  
 {  
 if( height < 72 )  
 {  
 System.out.println( "You are medium" );  
 }  
 else  
 {  
 System.out.println( "You are tall" );  
 }  
 }

1. When there is only one statement inside the braces of an IF or ELSE block, then the braces are not needed. Thus, this code

if( height < 72 )  
 {  
 System.out.println( "You are medium" );  
 }  
 else  
 {  
 System.out.println( "You are tall" );  
 }

could be written like this:

if( height < 72 )  
 System.out.println( "You are medium" );  
 else  
 System.out.println( "You are tall" );

or this:

if( height < 72 ) System.out.println( "You are medium" );  
 else System.out.println( "You are tall" );

Many times, I use the very first approach, including the braces. The reason is that often we find ourselves adding code later inside an IF or ELSE block. Then, it is convenient that the matching braces are already there.

1. An IF (or IF/ELSE) statement is considered one statement. This is true even if the IF (or ELSE) block contains multiple statements. For instance, originally, we had:

|  |
| --- |
| **Original** |
|  |
| if( height < 62 ) {  System.out.println( "You are not short" ); } else {  if( height < 72 )  {  System.out.println( "You are medium" );  }  else  {  System.out.println( "You are tall" );  } } |

Which we can rewrite in several ways:

|  |
| --- |
| **Rewrite 1** |
|  |
| if( height < 62 )  System.out.println( "You are short" ); else  if( height < 72 )   System.out.println( "You are medium" );  else   System.out.println( "You are tall" ); |

|  |
| --- |
| **Rewrite 2** |
|  |
| if( height < 62 )  System.out.println( "You are short" ); else if( height < 72 )   System.out.println( "You are medium" ); else   System.out.println( "You are tall" ); |

This second form is technically, an IF statement nested inside another IF statement. Look carefully at rewrite 1 and then compare it to rewrite 2. The only change is that two lines have been merged into one and the indention is changed on several others. We frequently refer to this second rewrite as an IF/ELSE-IF statement. More generally, we can write this statement as:

if( *booleanExpression1* )

{

// code

}

else if( *booleanExpression2* )

{

// code

}

else if( *booleanExpression3* )

{

// code

}

...

else

{

// code

}

**Boolean Operators**

1. A *boolean operator* takes boolean valued expressions as operands (or operand) and produces a boolean result. They are used to express *compound conditions*. For example, suppose that we want to print a message when someone whose height is more than 72 inches and whose weight is more than 220 pounds. We could use a nested IF statement to write that like this:

if( height > 72 )

{

if( weight > 220 )

{

System.out.println( "You are really big" );

}

}

Another way to write this using the boolean operator, *&&* is:

if( (height>72) && (weight>220) )  
 System.out.println( "You are really big" );

The *&&* operator produces a *true* result only when both the left operand (*e.g.* height>72) and the right operand (*e.g.* weight>220) are *true*. There are several other boolean operators that we will learn about.

**Conditional Expressions**

1. A conditional expression takes on a value depending on a boolean expression. It is similar to an IF/ELSE statement. Consider:

|  |  |  |
| --- | --- | --- |
| **IF/ELSE Technique** |  | **Conditional Expression Technique** |
|  |  |  |
| if( x > 10 )  y = 1; else  y = 2; |  | y = (x>10) ? 1 : 2; |

In other words, a conditional expression is

*boolean\_expression* ? *true\_result* : *false\_result*

The conditional expression doesn’t execute any statement. It simply yields one of the two results (in this case *1* or *2*). You *catch* the value it returns by setting a variable equal to the conditional expression. The conditional expression is an example of a *ternary operator* because it uses three operands. We generally only use these conditional expressions in cases where the result is fairly “simple.”

1. Examples:

char code = 'a'; boolean isSenior;

|  |  |  |
| --- | --- | --- |
| **IF/ELSE Technique** |  | **Conditional Expression Technique** |
|  |  |  |
| if( code == ‘s’ )  isSenior = true; else  isSenior = false; |  | isSenior = (code=='s') ? true : false; |

int w, x=23, y=5, z=7;

|  |  |  |
| --- | --- | --- |
| **IF/ELSE Technique** |  | **Conditional Expression Technique** |
|  |  |  |
| if( x < 10 )  w = y; else  w = z; |  | w = (x<10) ? y : z; |

**Formatting Console Output**

1. Format specifiers tell how to display variables to the console. Some example format specifiers:

|  |  |
| --- | --- |
| **Specifier** | **Meaning** |
| %5d | Displays an integer in at least 5 spaces, right-justified |
| %4.2f | Displays a floating point number with at least 4 total spaces, including the decimal point and two decimal values |
| %10s | Displays a string in at least 10 spaces, right justified |

1. Format specifiers are built into a format string which is then used in a special print method:

int height=73;  
 double weight = 212.359293284;  
   
 System.out.printf( "Ht = %3d\nWt = %6.2f", height, weight );

produces the output:

Ht = 73  
Wt = 212.36

**Switch Statement**

1. When we have just two alternatives, we use an IF/ELSE statement or a conditional expression. When we have more than two alternatives, we have seen that we can use nested IF statements (the IF/ELSE-IF statement). In this case, sometimes there is an alternative, the *switch* statement. The *switch* statement evaluates an integer (or char) expression and depending on its value executes the appropriate code.
2. Problem: A user will enter a floating point number and either a “1” to square the number or a “2” to cube the number.

int code;

// Read code and num from user.

switch ( code )  
{  
 case 1:   
 answer = Math.pow( num, 2 );  
 System.out.println( num + " squared is " + answer );  
 System.out.printf( "%10.2f squared is %10.2f \n",   
 num, answer );  
 break;   
   
 case 2:   
 answer = Math.pow( num, 3 );  
 System.out.printf( "%10.2f cubed is %10.2f \n",   
 num, answer );  
 break;  
  
 default:  
 System.out.printf(   
 "You didn't specify a valid operation for %8.4f \n", num );

}

1. The *break* statement is necessary (usually). If not present, code will continue to be executed until a break is found or the end of the switch statement occurs. For example, consider:

char year;  
   
 year = 'S';  
   
 switch( year )  
 {  
 case 'F': System.out.println("Freshman");  
 case 'S': System.out.println("Sophmore");  
 case 'J': System.out.println("Junior");  
 case 'E': System.out.println("Senior");  
 default : System.out.println("Unclassified Student");  
 }

which produces the output:

Sophmore  
Junior  
Senior  
Unclassified Student

**Operator Precedence**

1. When we have an expression we have to be concerned with the order in which operands are applied. Generally, we should always use parentheses to indicate order. However, it is still useful to understand operator precedence because not everyone may write code like you do; it is important to be able to understand other people’s code as well as write your own. This is a crude, first pass at understanding operator precedence:

|  |  |
| --- | --- |
| **Priority** | **Operator** |
| 1 | Increment & Decrement operators (x++, x--, ++x, --x) |
| 2 | Boolean operator: *not* (!) |
| 3 | Mathematics operators ( \*, /, % and then +, -) |
| 4 | Comparison operators (<, <=, >, >= and then ==, !=) |
| 5 | Boolean operators (^, then &&, then ||) |
| 6 | Assignment operators (=, +=, -=, \*=, /=, %=) |

There is a little bit more to it than this, but this is enough. Here is an example:

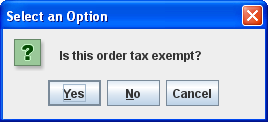
x=1; y=7;  
  
 if ( 2 \* y < 9 + 2 \* ++x )  
 System.out.println( "expression is true" );   
 else  
 System.out.println( "expression is false" );

which displays: *expression is false.*

**Confirmation Dialogs**

A confirmation dialog is a way to give the user a way to answer a question with “yes” or “no”, or the user can “cancel” the dialog.

double cost;  
   
cost = Double.parseDouble(  
 JOptionPane.showInputDialog("What is the cost of the order?") );  
   
int option = JOptionPane.showConfirmDialog(null,"Is order tax exempt?" );  
   
if( option == JOptionPane.YES\_OPTION )  
 System.out.printf( "Total cost is %6.2f", cost );  
else if( option == JOptionPane.NO\_OPTION )  
 System.out.printf( "Total cost is %6.2f", cost\*1.07 );  
else if( option == JOptionPane.CANCEL\_OPTION )  
 System.out.println( "You cancelled the dialog!" );  
else if( option == JOptionPane.CLOSED\_OPTION )  
 System.out.println( "You closed the dialog!" );

****

**PowerPoint**

Cover PowerPoint slides for Chapter 3 in class.

**Another Example**

A user will enter three integers. Display the message, “hit” if exactly 2 of the 3 possible sums of two numbers exceeds 50 and the product of all three numbers is not divisible by 3. Otherwise, display the message, “close” if the sum of the three integers is divisible by 3. Otherwise, display the message, “miss.”

**Boolean Operator Examples**

|  |  |
| --- | --- |
| 1. Consider this snippet of code:   boolean isXGreaterThanY;  int x = 3;  int y = 4;    if ( x > y )  {  isXGreaterThanY = true;  }  else  {  isXGreaterThanY = false;  }  System.out.println( isXGreaterThanY ); | 1. What is the output of this code? 2. Suppose that x=4 and y=4, what is the output? 3. Suppose that x=12 and y=5, what is the output? |

|  |  |
| --- | --- |
| 1. Consider this snippet of code:   boolean answer;  int x = 3;  int y = 4;  int z = 5;    if ( ( x > y ) && ( z > x ) )  {  answer = true;  }  else  {  answer = false;  }  System.out.println( answer ); | 1. What is the output of this code? 2. Suppose that x=4, y=3, z=5, what is the output? 3. Suppose that x=4, y=3, z=2, what is the output? |

|  |  |
| --- | --- |
| 1. Consider this snippet of code:   boolean answer;  int x = 3;  int y = 4;  int z = 5;    if ( ( x > y ) || ( z > x ) )  {  answer = true;  }  else  {  answer = false;  }  System.out.println( answer ); | 1. What is the output of this code? 2. Suppose that x=2, y=3, z=1, what is the output? |

|  |  |
| --- | --- |
| 1. Consider this snippet of code:   boolean answer;  int x = 3;  int y = 4;  int z = 5;    if ( ( x > y ) ^ ( z > x ) )  {  answer = true;  }  else  {  answer = false;  }  System.out.println( answer ); | 1. What is the output of this code? 2. Suppose that x=2, y=3, z=1, what is the output? 3. Suppose that x=2, y=1, z=3, what is the output? |

|  |  |
| --- | --- |
| 1. Consider this snippet of code:   boolean answer;    int x = 2;  int y = 1;  int z = 3;    if ( !( x > y ) && ( z > x ) )  {  answer = true;  }  else  {  answer = false;  }  System.out.println( answer ); | 1. What is the output of this code? 2. Suppose that x=2, y=3, z=1, what is the output? 3. Suppose that x=2, y=3, z=3, what is the output? |

|  |  |
| --- | --- |
| 1. Consider this snippet of code:   boolean a1, a2, a3, a4, a5;    int x = 2;  int y = 3;  int z = 3;    a1 = !( x > y ) || ( z > x );  a2 = !( x > y ) || !( z > x );  a3 = !( x > y ) && !( z > x );  a4 = !( x > y ) ^ !( z > x );  a5 = ( x > y ) ^ !( z > x );    System.out.println( a1 );  System.out.println( a2 );  System.out.println( a3 );  System.out.println( a4 );  System.out.println( a5 ); | What is the output of this code? |

1. Consider this snippet of code:

boolean a1, a2, a3, a4, a5, a6, a7, a8, a9, a10;

int x = 2;

int y = 4;

int z = 3;

a1 = ( ( x > y ) || ( z > x ) ) && ( z \* x < 10 );

a2 = ( ( x > y ) && ( z > x ) ) || ( z \* x < 10 );

a3 = ( ( x > y ) && ( z > x ) ) || !( z \* x < 10 );

a4 = ( z \* x > 9 ) || ( z - x < 1 ) || ( z >= y );

a5 = !( ( x < 1 ) && ( y > 1 ) );

a6 = ( z % x == 1 ) && ( x % z <= 2 );

a7 = a1 ^ a2;

a8 = ( a1 && a2 ) && ( !a3 && !a4 );

a9 = ( a1 || a3 ) && a2;

a10 = ( !a3 || a4 ) && a7;

System.out.println( a1 );

System.out.println( a2 );

System.out.println( a3 );

System.out.println( a4 );

System.out.println( a5 );

System.out.println( a6 );

System.out.println( a7 );

System.out.println( a8 );

System.out.println( a9 );

System.out.println( a10 )

What is the output of this code?

|  |  |
| --- | --- |
| 1. Consider this snippet of code:   if ( ( x == 1 ) || ( x++ < 10 ) )  {  System.out.println( “hey ” + x );  }  else  {  System.out.println( “you ” + x );  } | 1. Suppose that x=9, what is the output? 2. Suppose that x=1, what is the output? 3. Suppose that x=10, what is the output? |

|  |  |
| --- | --- |
| 1. Consider this snippet of code:   if ( ( x == 1 ) || ( --x < 10 ) )  {  System.out.println( “hey ” + x );  }  else  {  System.out.println( “you ” + x );  } | 1. Suppose that x=9, what is the output? 2. Suppose that x=1, what is the output? 3. Suppose that x=10, what is the output? |

1. Consider these three snippets of code. They all produce the same results.

|  |  |  |
| --- | --- | --- |
| Inefficient | Efficient | Efficient |
| if ( x < 60 )  grade = ‘F’;  else if ( ( x >= 60 ) && ( x < 70 ) )  grade = ‘D’;  else if ( ( x >= 70 ) && ( x < 80 ) )  grade = ‘C’;  else if ( ( x >= 80 ) && ( x < 90 ) )  grade = ‘B’;  else  grade = ‘A’; | if ( x < 60 )  grade = ‘F’;  else if ( x < 70 )  grade = ‘D’;  else if ( x < 80 )  grade = ‘C’;  else if ( x < 90 )  grade = ‘B’;  else  grade = ‘A’; | if ( x >= 90 )  grade = ‘A’;  else if ( x >= 80 )  grade = ‘B’;  else if ( x >= 70 )  grade = ‘C’;  else if ( x >= 60 )  grade = ‘D’;  else  grade = ‘F’; |

|  |  |
| --- | --- |
| 1. Consider this snippet of code:   boolean answer;  int x = 2;  int y = 3;  int z = 4;  int w;    if ( x <= y )  {  if ( x <= z )  w = x;  else  w = z;  }  else  {  if ( y <= z )  w = y;  else  w = z;  }  System.out.println( w ); | 1. What is the output of this code? 2. Suppose that x=2, y=4, z=3, what is the output? 3. Suppose that x=3, y=2, z=4, what is the output? 4. Suppose that x=4, y=2, z=3, what is the output? 5. Suppose that x=3, y=4, z=2, what is the output? 6. Suppose that x=4, y=3, z=2, what is the output? 7. Suppose that x=2, y=2, z=1, what is the output? 8. Suppose that x=2, y=2, z=3, what is the output? |