Quiz:

Which is the correct formatting in CS-152?

```java
public class Tmp
{
    public static void main(String[] args)
    {
        System.out.println("Pick Me");
    }
}
```

```java
public class Tmp
{
    public static void main(String[] args)
    {
        System.out.println("No, Me");
    }
}
```

```java
public class Tmp
{
    public static void main(String[] args)
    {
        System.out.println("Ooh, Ooh");
    }
}
```
The **if** Statement

The **if** statement tells your program to execute a certain section of code *only if* a particular test evaluates to **true**.

```java
if (grade >= 90) System.out.println("Cash Award");
```

When the “then” part is more than one statement, brackets are needed.

```java
if (grade >= 90)
{ System.out.println("Cash Award");
  cash = cash + 100.0;
}
```

Listing 3.1, Page 141 (with embellishments)

```java
import javax.swing.JOptionPane;
public class BankBalance
{
  private static final double OVERDRAWN_PENALTY = 8.00;
  private static final double INTEREST_RATE = 0.02; //2% annually
  public static void main(String[] args)
  {
    String str = JOptionPane.showInputDialog(null,
       "Enter Balance");

    double balance = Double.parseDouble(str);

    if (balance >= 0)
    { balance = balance + (INTEREST_RATE * balance) / 12;
    }
    else
    { balance = balance - OVERDRAWN_PENALTY;
    }
    JOptionPane.showMessageDialog(null,
       "balance after one month is $" + balance);
  }
}
import javax.swing.JOptionPane;

public class BankBalance
{
    public static final double OVERDRAWN_PENALTY = 8.00;
    public static final double INTEREST_RATE = 0.02; // 2% annually

    public static void main(String[] args)
    {
        String str = JOptionPane.showInputDialog(null, "Enter Balance");
        double balance = Double.parseDouble(str);
        if (balance >= 0)
        {
            balance = balance + (INTEREST_RATE * balance) / 12;
        }
        else
        {
            balance = balance - OVERDRAWN_PENALTY;
        }
        JOptionPane.showMessageDialog(null, "balance after one month is $" + balance);
    }
}

A very good way to study, is to **TYPE** the code in the textbook and/or slides into eclipse - and run.

- You will make mistakes, you will miss details such as ')' for '}' or ',' for '.'.
- This will help you learn to see such details.

**if-else** Statement Flow Chart

1) if (balance >= 0)
2) { balance = balance + (INTEREST * balance)/12;
3) }
4) else
5) { balance = balance - OVERDRAWN_PENALTY;
6) }
public class BankBalance
{
    public static final double OVERDRAWN_PENALTY = 8.00;
    public static final double INTEREST_RATE = 0.02;

    public static void main(String[] args)
    {

        In the code segment from listing 3.1 shown above, OVERDRAWN_PENALTY and INTEREST_RATE in are in all caps:

        a) because they are outside main, and, therefore the code would not actually compile.
        b) because they are declared static.
        c) because they are declared final.
        d) because they are global class variables.
        e) because later they are used in different branches of an if.

1) public class Toy_3_1
2) {
3)     public static void main(String[] args)
4)     {
5)         int x = 5;
6)     }
7)     if (x == (2+3)) x = x + 10;
8)     else x = x * 2;
9) }
10) System.out.println(x);
11) }
12) Output: 15

**if** and **else** Statements

*Either* statement 2 *or* statement 5 will execute. It is impossible for *both* to execute.

1) **if** (testscore >= 90)
2) { System.out.println("Great Job!");
3) }
4) **else**
5) { System.out.println("Work Harder");
6) }

Since the “then” part is only one statement, curly brackets are not needed.

However, when the code does not fit on a single line, the CS-152 coding standard dictates that the curly brackets be used.

---

**if, else if, else**

1) **public class** HelloWorld
2) { **public static void** main(String[] args)
3) {
4)     **int** x = 1;
5) }
6) **if** (x == 1) // ¡¡¡No semicolon!!!
7) { System.out.println("x is 1");
8) }
9) **else if** (x == 2) // ¡¡¡No semicolon!!!
10) { System.out.println("x is 2");
11) }
12) **else** // ¡¡¡No semicolon!!!
13) { System.out.println("x is special");
14) }
15) }
Use of `else if`

```java
public class HelloWorld {
    public static void main(String[] args) {
        int testscore = 76;
        char grade = 'F';

        if (testscore >= 90) grade = 'A';
        else if (testscore >= 80) grade = 'B';
        else if (testscore >= 70) grade = 'C';
        else if (testscore >= 60) grade = 'D';

        System.out.println("Grade = " + grade);
    }
}
```

With `testscore = 76`, the logical expressions in lines 8 and 9 would both evaluate to `true`. However, since 8 is `true`, execution never reaches 9.

Find the Syntax Error

```java
public class HelloWorld {
    public static void main(String[] args) {
        int testscore = 76;
        char grade;

        if (testscore >= 90) grade = 'A';
        else if (testscore >= 80) grade = 'B';
        else if (testscore >= 70) grade = 'C';
        else if (testscore >= 60) grade = 'D';

        System.out.println("Grade = " + grade);
    }
}
```

Local variable grade may not have been initialized.
else: Ensuring grade is Initialized

1) public class HelloWorld
2) { public static void main(String[] args)
3) { int testscore = 76;
4) char grade;
5) if (testscore >= 90) grade = 'A';
6) else if (testscore >= 80) grade = 'B';
7) else if (testscore >= 70) grade = 'C';
8) else if (testscore >= 60) grade = 'D';
9) else grade = 'F';
10) System.out.println("Grade = " + grade);
11) }
12) }
13)

The compiler recognizes there is no path to line 11 in which grade is not initialized.

Quiz: if and else if

1) int testscore = 88;
2) char grade = 'F';
3)
4) if (testscore >= 60) grade = 'D';
5) else if (testscore >= 70) grade = 'C';
6) else if (testscore >= 80) grade = 'B';
7) else if (testscore >= 90) grade = 'A';
8)
9) System.out.println(grade);

What would be the output of the above Java code?
a) B  b) D  c) F  d) FB  e) FDCB
Control Flow and Program State

1) public class HelloWorld
2) { public static void main(String[] args)
3) { int x = 5;
4) int a = 0;
5) if (x < 10) a=1;
6) if (x < 6) a=2;
7) if (x < 1) a=3;
8) System.out.println(a);
9) }
10) }

Table of program state at the start of each line in the order of execution.

Output: 2
Control Flow: \textit{if} \& \textit{else if}

1) \texttt{public static void main(String[] args)}
2) {
3) \hspace{1em} \texttt{int x = 5;}
4) \hspace{1em} \texttt{int a = 0;}
5) \hspace{1em} \texttt{if (x < 10)}
6) \hspace{2em} \texttt{\{ a=1; \}}
7) \hspace{1em} \texttt{\}}
8) \hspace{1em} \texttt{\} else if (x < 6)}
9) \hspace{2em} \texttt{\{ a=2; \}}
10) \hspace{1em} \texttt{\}}
11) \hspace{1em} \texttt{\} else if (x < 1)}
12) \hspace{2em} \texttt{\{ a=3; \}}
13) \hspace{1em} \texttt{\}}
14) \texttt{System.out.println(a);}
15) \texttt{)}

\begin{tabular}{|l|c|c|}
\hline
start of line & x & a \\
\hline
2) & 5 & \\
3) & 5 & 0 \\
4) & 5 & 0 \\
5) & 5 & 1 \\
13) & 5 & 1 \\
\hline
\end{tabular}

Output: 1

Quiz: \textit{if, else if, else}

1) \texttt{public static void main(String[] args)}
2) {
3) \hspace{1em} \texttt{int x = 1;}
4) \hspace{1em} \texttt{if (x == 1)}
5) \hspace{2em} \texttt{\{ System.out.println("x is 1"); \}}
6) \hspace{1em} \texttt{\} else if (x == 2)}
7) \hspace{2em} \texttt{\{ System.out.println("x is 2"); \}}
8) \hspace{1em} \texttt{\}}
9) \hspace{1em} \texttt{\} else x = 3;}
10) \hspace{1em} \texttt{\{ System.out.println("wild: x=" + x); \}}
11) \hspace{1em} \texttt{\}}
12) \texttt{)}

\begin{tabular}{|l|l|}
\hline
The output is: & a) x is 1 \quad b) x is 1 \quad c) x is 2 \quad d) wild: x=1 \quad e) wild: x=3 \\
& wild: x=1 \\
\hline
\end{tabular}

Look carefully
This code does not do what it was probably intended to do.
### if, else if, else

```java
1) public static void main(String[] args) {
2)   int x = 4;
3)   if (x == 1) {
4)     System.out.println("x is 1");
5)   } else if (x == 2) {
6)     System.out.println("x is 2");
7)   } else x = 3;
8)   System.out.println("x is " + x);
9) }
```

**Output:**
```
x is 3
```

### Example: if, else if, else

```java
1) int x=3, y=7;
2) if (x*x < x+y) {
3)   System.out.println("B");
4) }
5) if (x > 0) {
6)   System.out.println("E");
7) }
8) else if (y > 0) {
9)   System.out.println("A");
10) }
11) else {
12)   System.out.println("T");
13) }
14) if (x*y > 0) {
15)   System.out.println("S");
16) }
```

**Output:**
```
BES
```
Quiz: if & else if

1) public static void main(String[] args)
2) {
3)   int x = 50;
4)   if (x > 20)
5)     { System.out.print("A");
6)     }
7)   else if (x > 30)
8)     { System.out.print("B");
9)     }
10)  else if (x > 40)
11)  { System.out.print("C");
12)  }
13)  System.out.println("D");
14) }

The output is:
- a) ABCD
- b) ABD
- c) CD
- d) C
- e) AD

Quiz: if, else if

1) int a = 3, b = 4, c = 5;
2)
3) if (a+b > c) System.out.print("A");
4)    else if (a*b > c) System.out.print("B");
5)    if (a*a > c) System.out.print("C");
6)    else if (b*b > c) System.out.print("D");
7)    else if (b*b >= c*a) System.out.print("E");
8)    System.out.println("F");

The output of this Java code segment is:
- a) AF
- b) ACF
- c) ABCF
- d) ACDF
- e) ACDEF
Logical Operators

== Equals
!= Not Equal
< Less than
> Greater than
<= Less than or Equal to
>= Greater than or Equal to

| Logical OR
&& Logical AND
! Logical NOT

The bitwise operators are not covered in CS-152.

| bitwise OR
& bitwise AND
^ bitwise exclusive OR
~ bitwise NOT

Truth Tables for AND, OR and NOT

1) public class TruthTables
2) {
3)   public static void main(String[] args)
4)   {
5)     System.out.println( true && true );
6)     System.out.println( true && false );
7)     System.out.println( false && true);
8)     System.out.println( false && false );
9)     System.out.println( true || true );
10)    System.out.println( true || false );
11)    System.out.println( false || true );
12)    System.out.println( false || false );
13)    System.out.println(!true);
14)    System.out.println(!false);
15)    System.out.println(!true);
16)    System.out.println(!false);
17) }
18) }
The Logical AND Operator: \&\&

true when both parts are true

1) \textbf{if} ( (pressure >= min) \&\& (pressure <= max) )
2) \{ System.out.println("Pressure is OK");
3) \}
4) \textbf{else}
5) \{ System.out.println(
6) \quad "Warning: Pressure is out of range.");
7) \}

The order of operations, in \textit{Java}, makes these equivalent:
\textbf{if} ( (pressure >= min) \&\& (pressure <= max) )
\textbf{if} ( pressure >= min \&\& pressure <= max )

The Logical AND Operator: | |

true when both parts are true
true when either part is true

1) \textbf{if} ( (pressure < min ) || (pressure > max ) )
2) \{ System.out.println(  
3) \quad "Warning: Pressure is out of range.";
4) \}
5) \textbf{else}
6) \{ System.out.println("Pressure is OK");
7) \}
What Happens on Line 4?

1) public class TruthTable
2) { public static void main(String[] args)
3) {
4) System.out.println(xor(true, true));
5) System.out.println(xor(true, false));
6) System.out.println(xor(false, true));
7) System.out.println(xor(false, false));
8) }
9)
10) public static boolean xor(boolean a, boolean b)
11) {
12) return ( a || b) && !(a && b);
13) }
14) }

15) Truth Table for Exclusive OR

Trace of What Happens on Line 5

1) public class TruthTable
2) { public static void main(String[] args)
3) {
4) System.out.println(xor(true, true));
5) System.out.println(xor(true, false));
6) System.out.println(xor(false, true));
7) System.out.println(xor(false, false));
8) }
9)
10) public static boolean xor(boolean a, boolean b)
11) {
12) return ( a || b) && !(a && b);
13) }
14) }

15) Truth Table for Exclusive OR
Which of the \texttt{if} statements have identical truth tables?

1) \texttt{public class TruthTable}
2) \{ \texttt{public static void main(String[]} \texttt{args)}
3) \{ \texttt{foo(true,true); \hspace{1cm} CAT DOG}
4) \texttt{foo(true,false); \hspace{1cm} CAT BAT DOG}
5) \texttt{foo(false,true); \hspace{1cm} CAT BAT DOG}
6) \texttt{foo(false,false); \hspace{1cm} ANT BAT}
7) \}
8) \}
9) \texttt{public static void foo(boolean a, boolean b)}
10) \{ \texttt{if ( a || b) System.out.print("CAT ");}
11) \texttt{if ( !a && !b) System.out.print("ANT ");}
12) \texttt{if (! ( a && b)) System.out.print("BAT ");}
13) \texttt{if ( !( !a && !b)) System.out.print("DOG ");}
14) \texttt{System.out.println();}
15) \}
16) \}
17)
18)\}

"Short circuit" Evaluation

What is meant by “this expression is safe”.
\[
\texttt{if (x != 0.0 && 1.0/x > x + y)}
\]

The expression: \texttt{(exp1 && exp2)}
can only be \texttt{true} if both \texttt{exp1} and \texttt{exp2} are \texttt{true}.
Thus, if \texttt{exp1} is \texttt{false}, Java does not evaluate \texttt{exp2}.
Thus, if \texttt{x = 0}, then \texttt{1.0/x} is not evaluated.
Order of Evaluation

```java
double x = 2.0;
double y = -2.0;
if (x != 0.0 && 1.0/x > x + y)
{
    1) x != 0.0
    2) 1.0/x
    3) x + y
    4) 0.5 > 0.0
    5) true && true
    6) true
}
```

Comparing Strings

```java
String myColor = 
    JOptionPane.showInputDialog(null,
    "Enter a color");
System.out.println(myColor == "red");
System.out.println(myColor.equals("red"));
```
Comparing Addresses verses Strings

```java
String myColor;
//Some code that puts data in myColor.

myColor == "red"
true if and only if the memory location of myColor is the same as the memory location of "red"

myColor.equals("red")
true if and only if the data in myColor is "red".
```

Possible Null Pointer Exception

```java
1) String str = JOptionPane.showInputDialog(null, "Enter red");
2) if (str.equals(""))
3) { System.out.println("Clicked OK with no data");
4) }
5) if (str == null)
6) { System.out.println("Clicked Cancel");
7) }
```

A Null Pointer Exception occurs when code attempts to access a member of an object using an object reference that is not pointing anywhere.

In line 3, equals is a method that is a member of a String object.

However, if str is null, then, even though str was declared as being a reference to a String object, it doesn't, in fact, point to anything.
Does This Fix the Problem?

```java
1) String str = JOptionPane.showInputDialog(null,
2) "Enter red");
3) if (str == null)
4) { System.out.println("Clicked Cancel");
5) }
6) if (str.equals(""))
7) { System.out.println("Clicked OK with no data");
8) }
```

Checking Input: All Golden

```java
1) String str = JOptionPane.showInputDialog(
2) null, "Enter red");
3) if (str == null)
4) { System.out.println("Clicked Cancel");
5) }
6) else if (str.equals(""))
7) { System.out.println("OK with no data");
8) }
9) String str = JOptionPane.showInputDialog(
10) null, "Enter red");
11) if (str == null)
12) { System.out.println("Clicked Cancel");
13) System.exit(0);
14) }
15) if (str.equals(""))
16) { System.out.println("OK with no data");
17) }
18) ```
How can this be coded in Java?

- Let X be a course with 3 grades: a midterm, a final exam and a final project. Each counts as 1/3 of the course grade.
  Example: \( t_1 = 90, t_2 = 70, p = 80 \rightarrow \text{grade} = 80.0 \)

- However, if a student does better on the final than on the midterm, then the final is counted with twice the weight as the midterm.
  Example: \( t_1 = 0, t_2 = 90, p = 90 \rightarrow \text{grade} = 70.0 \)

- A student who gets a 95% or more on the final project can drop his or her final exam score and count the final project as 2/3 and the midterm as 1/3 of the course grade.
  Example: \( t_1 = 90, t_2 = 0, p = 90 \rightarrow \text{grade} = 90.0 \)

Example Calculation: Count Final 2x Midterm

**Case 1**

\[
\text{grade} = \left( t_1 + t_2 + p \right)/3
\]

**Case 2**

Counting \( t_2 \) with twice the weight as \( t_1 \)

**Given:**

\[
\text{grade} = (w)t_1 + (2w)t_2 + (1/3)p
\]

\[
w + 2w + 1/3 = 1
\]

\[
w + 2w = 2/3
\]

\[
3w = 2/3
\]

\[
w = 2/9
\]

\[
\text{grade} = (2/9)t_1 + (4/9)t_2 + (1/3)p
\]

Paper & Pencil **before** code!
Java Code for Conditional Grade

```java
Scanner in = new Scanner(System.in);
double t1 = in.nextDouble();
double t2 = in.nextDouble();
double p = in.nextDouble();

double grade = (t1 + t2 + p) / 3.0;

if (t2 > t1) {
    grade = t1*(2.0/9.0) + t2*(4.0/9.0) + p/3.0;
}

if (p >= 95) {
    double tmp = (t1/3.0) + (p*2.0/3.0);
    if (tmp > grade) grade = tmp;
}

System.out.println("Grade="+grade);
```