Conditionals
1. If-then-else

A conditional is a test: ‘if something is true, then do this, otherwise maybe do something else’. The C++ syntax is

```cpp
if ( something ) {
    // do something;
} else {
    // do otherwise;
}
```

- The ‘else’ part is optional
- You can leave out braces in case of single statement.
2. Complicated conditionals

Chain:

```java
if ( /* some test */ ) {
    ...
} else if ( /* other test */ ) {
    ...
}
```

Nest:

```java
if ( /* some test */ ) {
    if ( /* other test */ ) {
        ...
    } else {
        ...
    }
} else {
    ...
}
```
3. What are logical expressions?

```plaintext
logical_expression ::
    comparison_expression
    | NOT comparison_expression
    | logical_expression CONJUNCTION comparison_expression

comparison_expression ::
    numerical_expression COMPARE numerical_expression

numerical_expression ::
    quantity
    | numerical_expression OPERATOR quantity

quantity :: number | variable
```
4. Comparison and logical operators

Here are the most common logic operators and comparison operators.

<table>
<thead>
<tr>
<th>Operator</th>
<th>meaning</th>
<th>example</th>
</tr>
</thead>
<tbody>
<tr>
<td>==</td>
<td>equals</td>
<td>x==y-1</td>
</tr>
<tr>
<td>!=</td>
<td>not equals</td>
<td>x*x! = 5</td>
</tr>
<tr>
<td>&gt;</td>
<td>greater</td>
<td>y &gt; x - 1</td>
</tr>
<tr>
<td>&gt;=</td>
<td>greater or equal</td>
<td>sqrt(y) &gt;= 7</td>
</tr>
<tr>
<td>&lt;,&lt;=</td>
<td>less, less equal</td>
<td>x &lt; 1 &amp;&amp; x &gt; 0</td>
</tr>
<tr>
<td>&amp;&amp;,</td>
<td></td>
<td></td>
</tr>
<tr>
<td>and, or</td>
<td>x &lt; 1 and x &gt; 0</td>
<td>!( x &gt; 1 &amp;&amp; x &lt; 2 )</td>
</tr>
<tr>
<td>!</td>
<td>not</td>
<td>not ( x &gt; 1 and x &lt; 2 )</td>
</tr>
</tbody>
</table>

*Precedence* rules of operators are common sense. When in doubt, use parentheses.
Exercise 1

The following code claims to detect if an integer has more than 2 digits.

Code:
```cpp
int i;
cin >> i;
if ( i>100 )
    cout << "That number " << i << " had more than 2 digits" << endl;
```

Output
[basic] if:
```
... with 50 as input
....
... with 150 as input
....
That number 150 had more than 2 digits
```

Fix the small error in this code. Also add an ‘else’ part that prints if a number is negative.

*You can base this off the file if.cxx in the repository*
Review quiz 1

True or false?

- The tests `if (i>0)` and `if (0<i)` are equivalent.
  /poll "Same tests: 'i>0' and '0<i'?" "T" "F"

- The test

  ```cpp
  if (i<0 && i>1)
      cout << "foo"
  ```

  prints `foo` if `i < 0` and also if `i > 1`.
  /poll "'if (i<0 && i>1)' is true if i negative and if i greater than one" "T" "F"

- The test

  ```cpp
  if (0<i<1)
      cout << "foo"
  ```

  prints `foo` if `i` is between zero and one.
  /poll "'if (0<i<1)' true if i between 0 and 1" "T" "F"
Review quiz 2

Any comments on the following?

```cpp
bool x;
// ... code with x ...
if ( x == true )
    // do something
```
Exercise 2

Read in an integer. If it is even, print ‘even’, otherwise print ‘odd’:

```cpp
if ( /* your test here */ )
    cout << "even" << endl;
else
    cout << "odd" << endl;
```

Then, rewrite your test so that the true branch corresponds to the odd case.
Exercise 3

Read in a positive integer. If it’s a multiple of three print ‘Fizz!’; if it’s a multiple of five print ‘Buzz!’. If it is a multiple of both three and five print ‘Fizzbuzz!’. Otherwise print nothing.

Note:

• Capitalization.
• Exclamation mark.
• Your program should display at most one line of output.
Turn it in!

- If you have compiled your program, do:
  
  `coe_fizzbuzz yourprogram.cc`

  where ‘yourprogram.cc’ stands for the name of your source file.

- Is it reporting that your program is correct? If so, do:
  
  `coe_fizzbuzz -s yourprogram.cc`

  where the -s flag stands for ‘submit’.

Note: this will send your file to the instructors with a time stamp. If you submit again after the deadline, you will be recorded as a late submission.
Project Exercise 4

Read two numbers and print a message stating whether the second is a divisor of the first:

Code:

```cpp
int number, divisor;
bool is_a_divisor;
/* ... */
if ( /* ... */
    /* ... */
) {
    cout << "Indeed, " << divisor
         << " is a divisor of "
         << number << endl;
} else {
    cout << "No, " << divisor
         << " is not a divisor of "
         << number << endl;
}
```

Output

[primes] division:

( echo 6 ; echo 2 ) |
1divisiontest
Enter a number:
Enter a trial divisor:
Indeed, 2 is a divisor of 6

( echo 9 ; echo 2 ) |
1divisiontest
Enter a number:
Enter a trial divisor:
No, 2 is not a divisor of 9
5. Switch statement example
Cases are executed consecutively until you ‘break’ out of the switch statement:

Code:

```cpp
switch (n) {
    case 1 :
    case 2 :
        cout << "very small" << endl;
        break;
    case 3 :
        cout << "trinity" << endl;
        break;
    default :
        cout << "large" << endl;
}
```

Output

```
[basic] switch:
for v in 1 2 3 4 5 :
    do \
        echo $v |
    ./switch ; \
    done
very small
very small
trinity
large
large
```
6. Local variables in conditionals

The curly brackets in a conditional allow you to define local variables:

```java
if ( something ) {
    int i;
    .... do something with i
}
// the variable ‘i’ has gone away.
```

Good practice: only define variable where needed.

Braces induce a scope.