Conditionals

Victor Eijkhout, Susan Lindsey

Fall 2022

last formatted: March 27, 2023

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

```
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:

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

Nest:

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

3. Comparison and logical operators

Here are the most common logic operators and comparison operators:

Operator	meaning	example
==	equals	`x==y-1`
!=	not equals	`x*x!=5`
>	greater	`y>x-1`
>=	greater or equal	`sqrt(y)>=7`
<,<=	less, less equal	
&&,||	and, or	`x<1 && x>0`
and,or	and, or	`x<1 and x>0`
!	not	`!( x>1 && x<2 )`
not		`not ( x>1 and x<2 )`

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:

```
1 int i;
2 cin >> i;
3 if ( i>100 )
4   cout << "That number " << i
5        << " had more than 2 digits"
6        << '\n';
```

Output:

```
... 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

  ```
  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

  ```
  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?

```
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’:

```
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!'. 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 as divisor of the first:

Code:

```
1   int number,divisor;
2   bool is_a_divisor;
3   /* ... */
4   if (
5   /* ... */
6       ) {
7     cout << "Indeed, " << divisor
8          << " is a divisor of "
9          << number << '\n';
10  } else {
11    cout << "No, " << divisor
12         << " is not a divisor of "
13         << number << '\n';
14  }
```

Output:

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

( echo 9 ; echo 2 ) |
     divisiontest
Enter a number:
Enter a trial divisor:
No, 2 is not a divisor
     of 9
```

4. Switch statement example

Cases are executed consecutively until you 'break' out of the switch statement:

Code:

```
1 switch (n) {
2 case 1 :
3 case 2 :
4   cout << "very small" << '\n';
5   break;
6 case 3 :
7   cout << "trinity" << '\n';
8   break;
9 default :
10  cout << "large" << '\n';
11 }
```

Output:

```
for v in 1 2 3 4 5 ; do
    \
        echo $v |
    ./switch ; \
      done
very small
very small
trinity
large
large
```

5. Local variables in conditionals

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

```
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.