Conditionals

Victor Eijkhout, Susan Lindsey

COE 322 Fall 2020

Conditionals

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.

Complicated conditionals

Chain:

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

Nest:

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

What are logical expressions?

```
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
```

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`		`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.

Review quiz 1

True or false?

- The tests `if (i>0)` and `if (0<i)` are equivalent.
- The test

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

  prints `foo` if $i < 0$ and also if $i > 1$.
- The test

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

  prints `foo` if i is between zero and one.

Any comments on the following?

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

Exercise 1

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 2

Read in a positive integer. If it's a multiple of three print 'Fizz!'; if it's a multiple of five print 'Buzz'!. It it is a multiple of both three and five print 'Fizzbuzz!'. Otherwise print nothing. (Note: your program should display at most one 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 3

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

Code:

```
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 ) | 1division
Enter a number:
Enter a trial divisor:
Indeed, 2 is a divisor of 6

( echo 9 ; echo 2 ) | 1division
Enter a number:
Enter a trial divisor:
No, 2 is not a divisor of 9
```

Switch statement example

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

Code:

```
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
```

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.