#### Conditionals

Kevin Schmidt, Susan Lindsey, Charlie Dey

Spring 2019



#### **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 ( something ) {
} else if ( something else ) {
Nest:
if ( something ) {
  if ( something else ) {
  } else {
```



# What are logical expressions?

```
logical_expression ::
   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**

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 )

*Precendence* rules are common sense. When in doubt, use parentheses.



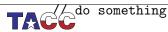
#### Review quiz 1

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.</li>

Any comments on the following?

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



## 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 :</pre>

cout << "large" << endl;</pre>

# 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.
```



#### Exercise 1

Read in an 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.



# **Project Exercise 2**

Read two numbers and print a message like

3 is a divisor of 9

if the first is an exact divisor of the second, and another message

4 is not a divisor of 9

if it is not.

