

# Conditionals

Victor Eijkhout, Susan Lindsey

Fall 2024

last formatted: August 28, 2024

# 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	<code>x==y-1</code>
!=	not equals	<code>x*x!=5</code>
>	greater	<code>y&gt;x-1</code>
>=	greater or equal	<code>sqrt(y)&gt;=7</code>
<,<=	less, less equal	
&&,	and, or	<code>x&lt;1 &amp;&amp; x&gt;0</code>
and,or	and, or	<code>x&lt;1 and x&gt;0</code>
!	not	<code>!( x&gt;1 &amp;&amp; x&lt;2 )</code>
not		<code>not ( x&gt;1 and x&lt;2 )</code>

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

# Quiz 1

Any comments on the following?

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

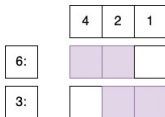
# Exercise 1

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.

## 4. Bitwise operations



Code:

```
// basic/bitor.cpp
int x=6,y=3;
cout << "6|3 = " << (x|y)
      << '\n';
cout << "6&3 = " << (x&y)
      << '\n';
```

Output:

```
6|3 = 7
6&3 = 2
```

## Exercise 2

How would you test if a number is odd or even with bitwise testing?



## 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.

## 6. Conditional with initializer

Variable local to the conditional:

Code:

```
// basic/ifinit.cpp
if ( char c = getchar();
    c!='a' )
    cout << "Not an a, but: "
          << c
          << '\n';
else
    cout << "That was an a!"
          << '\n';
```

Output:

```
Script:
for c in d b a z ;
do \
echo $c |
./ifinit ; \
done
Not an a, but: d
Not an a, but: b
That was an a!
Not an a, but: z
```

## Exercise 3

Write a function `float read_number()` and use it to rewrite your fizzbuzz solution.

Make sure to use an initializer; the number you are testing should be limited in scope to the conditional.

## 7. Switch statement example

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

Code:

```
// basic/switch.cpp
switch (n) {
case 1 :
case 2 :
    cout << "very small"
        << '\n';
    break;
case 3 :
    cout << "trinity" <<
        '\n';
    break;
default :
    cout << "large" <<
        '\n';
}
```

Output:

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