LOGICAL OPERATIONS, CONTROL STRUCTURES

Today

- The `if` statement
- Relational operators
- Logical operators
- Truth tables

the `if` branching statement

- We have already been using the if statement:

```
void goNorth()
{
// Stop the ant going over the end of the world

   if ( y < 11)
   {
     y = y + 1;
   }
}
```

- Let's look at it in a bit more detail.

The `if` branching statement

- The `if` is a *conditional*
  – Means the computer makes a choice
- It is also a *control structure*
- General structure:

```
if(<something that is true or false>)
{

    <some instructions>

}
```

Boolean expressions

- Boolean expressions are things that are true or false.
- Boolean variables: `true` (1) or `false` (0)
- Logical operators:

!	not
&&	and
||	or

- Example:

```
boolean a, b;
x = 1; // true
y = 0; // false

   if(x && y)
   {
     cout << ''This is false\n'';
   }

   if(x || y)
   {
     cout << ''This is true\n'';
   }
```

Truth tables

a	!a
false	true
true	false

a	b	a && b
true	true	true
true	false	false
false	true	false
false	false	false

a	b	a || b
true	true	true
true	false	true
false	true	true
false	false	false

Relational operators

==	equality
!=	inequality
>	greater than
<	less than
>=	greater than or equal to
<=	Less than or equal to

example:

```
int x, y;
x = -5;
y = 7;
```

some truths:

(x < y)	true
(x == y)	false
(x >= y)	false

The `if` branching statement (again)

```
// Is the ant still in the world?

if ((x < 10) && (y < 10))
{
  cout << ''The ant didn't fall off the world\n'';
}
```

The `if-else` branching statement

- A neater way of doing some branching.
- This:

```
// goNorth and wrap around

if (y < 10)
{
  y = y + 1;
}

if (y == 10)
{
  y = 0;
}
```

- Is a bit neater as:

```
// goNorth and wrap around

   if (y < 10)
   {
     y = y + 1;
   }
   else
   {
     y = 0;
   }
```

- General structure:

```
if(<something that is true or false>)
{

    <some instructions>

}
else
{
    <alternative instructions>
}
```

the `while` looping statement.

- `while` allows us to repeat things:

```
// goToNorthernEdge

   while (y <= 10)
    {
      y = y + 1;
    }
```

- General structure:

```
while(<something that is true or false>)
{

    <some instructions>

}
```

- This structure looks a lot like `if`

Summary

- We covered some of the basic control structures:
  - `if`, `while`
- Along the way we looked at boolean expressions and relational operators as well.