- Previous Lecture (and lab):
  - Variables & assignment
  - Built-in functions
  - Input & output
  - Good programming style (meaningful variable names; use comments)
- Today's Lecture:
  - Branching (conditional statements)

Announcements:

- Discussion section this week in Hollister 464 computer lab
- Project 1 (P1) due Thurs, 9/1, at 11pm
- Pay attention to Academic Integrity
- You can see any TA for help, not just your discussion TA
- Matlab consultants at ACCEL Green Rm (Carpenter Hall 2nd fl. computing facility) 5-10pm Sunday to Thursday
- Piazza – "Q & A system" for all students in CS1112. Use it for clarification only—do not ask (answer) homework questions and do not give hints on homework. Will be monitored by TAs.
- Please register your clicker using the link on the course website (redirected to Cornell IT)—not through Blackboard
- Remote MATLAB access: newly joined students will have accounts tomorrow

Quick review

- Variable
  - A named memory space to store a value
- Assignment operator: =
  - Let x be a variable that has a value. To give variable y the same value as x, which statement below should you write?

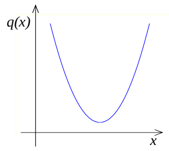
    x = y   or   y = x
- Script (program)
  - A sequence of statements saved in an m-file
- ; (semi-colon)
  - Suppresses printing of the result of assignment statement

- So far, all the statements in our scripts are executed in order
- We do not have a way to specify that some statements should be executed only under some condition
- We need a new language construct…

Consider the quadratic function

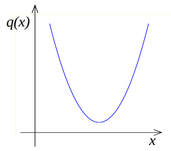
$$q(x) = x^2 + bx + c$$

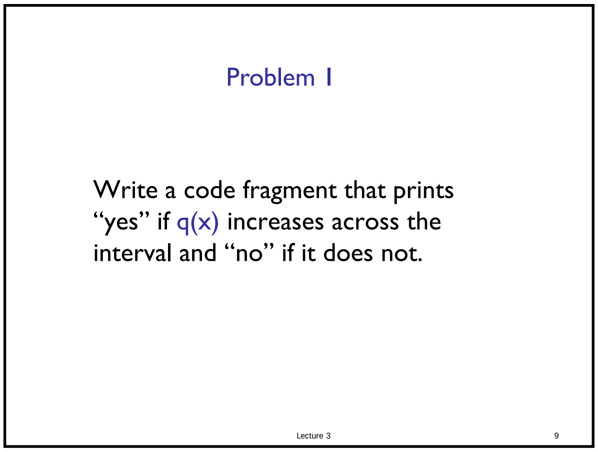
on the interval $[L\ ,\ R]$:



- Is the function strictly increasing in $[L\ ,\ R]$?
- Which is smaller, $q(L)$ or $q(R)$?
- What is the minimum value of $q(x)$ in $[L\ ,\ R]$?

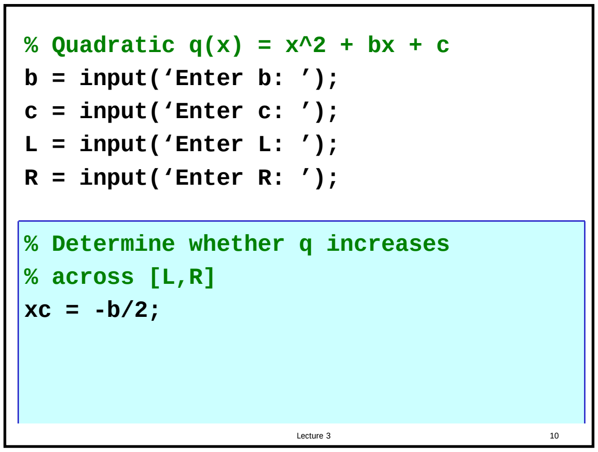
- What are the critical points?
  - End points: $x = L$, $x = R$
  - $\{\ x \mid q'(x) = 0\ \}$



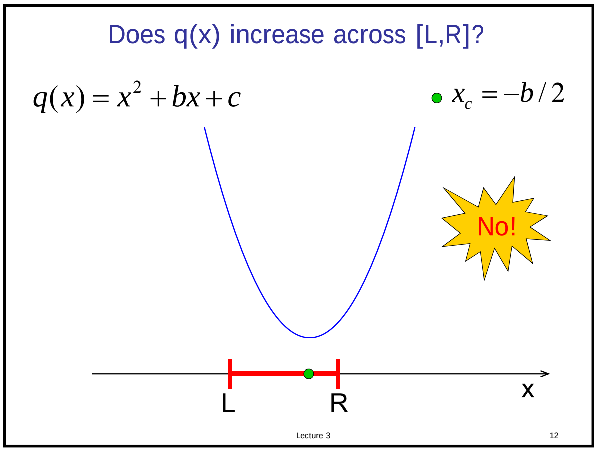


Problem 1

Write a code fragment that prints "yes" if q(x) increases across the interval and "no" if it does not.



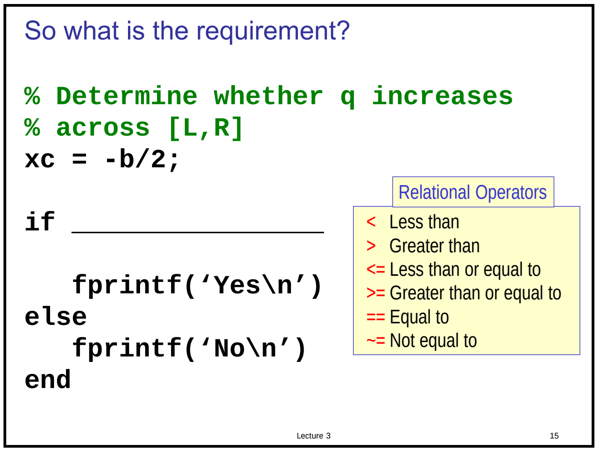
```
% Quadratic q(x) = x^2 + bx + c
b = input('Enter b: ');
c = input('Enter c: ');
L = input('Enter L: ');
R = input('Enter R: ');

% Determine whether q increases
% across [L,R]
xc = -b/2;
```



Does q(x) increase across [L,R]?

$q(x) = x^2 + bx + c$

$x_c = -b/2$

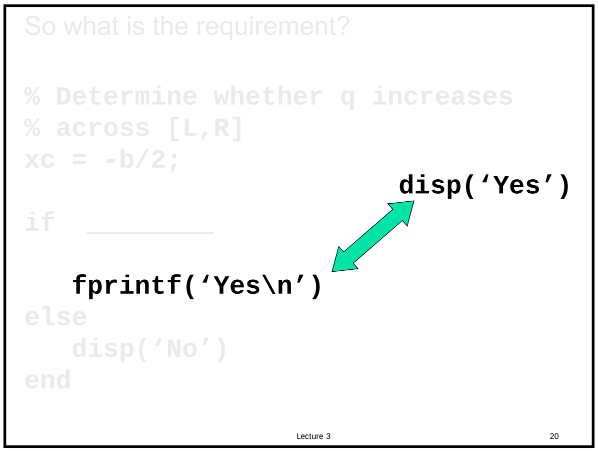


So what is the requirement?

```
% Determine whether q increases
% across [L,R]
xc = -b/2;

if ______________

   fprintf('Yes\n')
else
   fprintf('No\n')
end
```

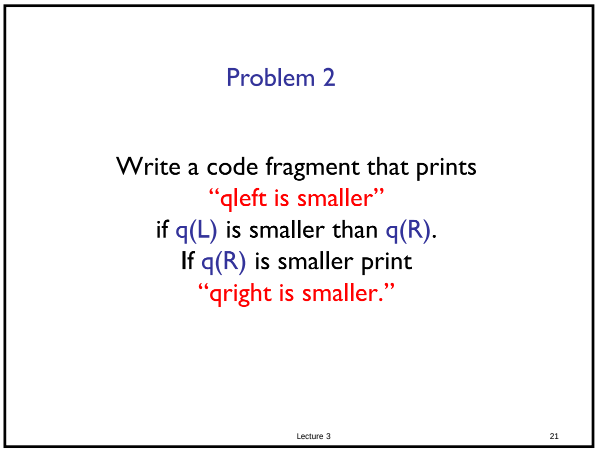
Relational Operators

<	Less than
>	Greater than
<=	Less than or equal to
>=	Greater than or equal to
==	Equal to
~=	Not equal to



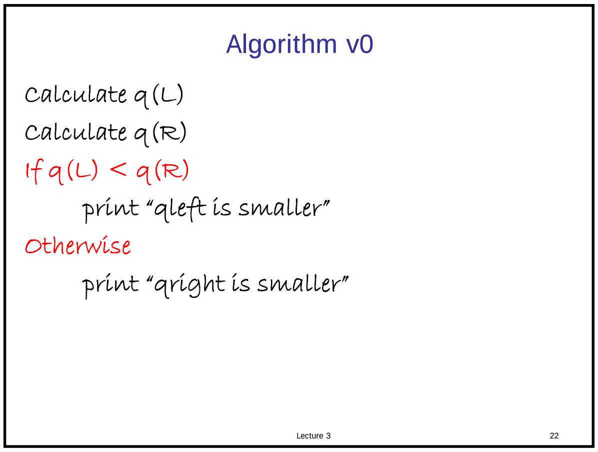
So what is the requirement?

```
% Determine whether q increases
% across [L,R]
xc = -b/2;
                    disp('Yes')
if ________

   fprintf('Yes\n')
else
   disp('No')
end
```



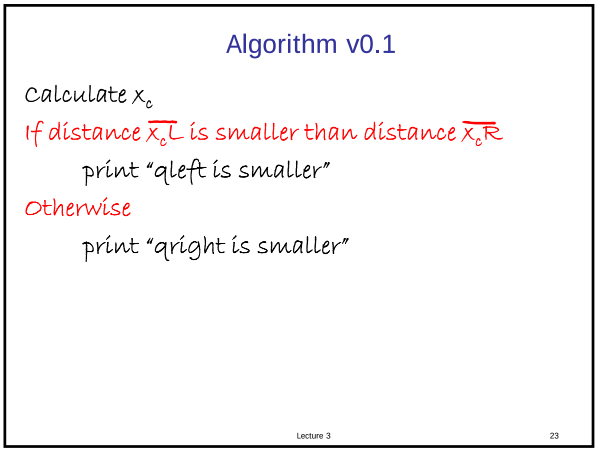
Problem 2

Write a code fragment that prints "qleft is smaller" if q(L) is smaller than q(R). If q(R) is smaller print "qright is smaller."



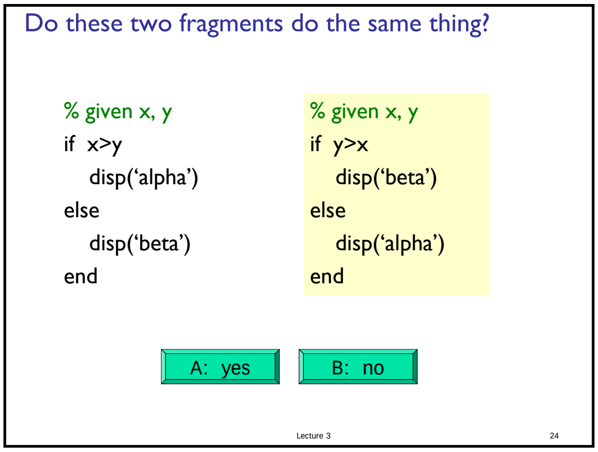
Algorithm v0

```
Calculate q(L)
Calculate q(R)
If q(L) < q(R)
     print "qleft is smaller"
Otherwise
     print "qright is smaller"
```



Algorithm v0.1

```
Calculate $x_c$
If distance $\overline{x_cL}$ is smaller than distance $\overline{x_cR}$
     print "qleft is smaller"
Otherwise
     print "qright is smaller"
```

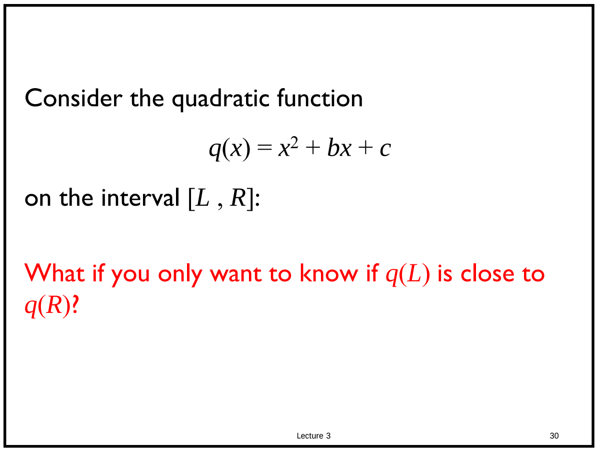


Do these two fragments do the same thing?

```
% given x, y
if  x>y
   disp('alpha')
else
   disp('beta')
end
```

```
% given x, y
if  y>x
   disp('beta')
else
   disp('alpha')
end
```

A: yes B: no

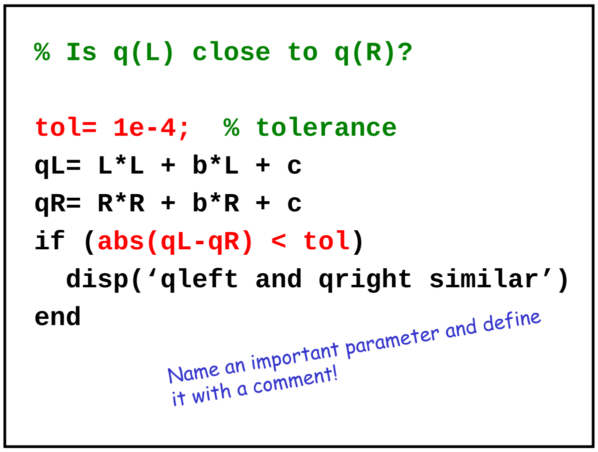


Consider the quadratic function

$$q(x) = x^2 + bx + c$$

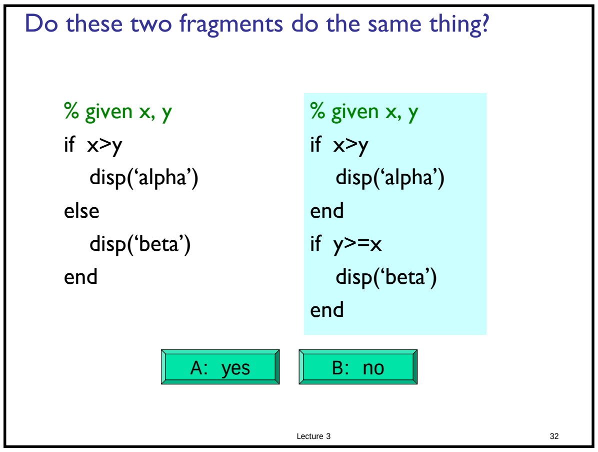
on the interval $[L , R]$:

What if you only want to know if $q(L)$ is close to $q(R)$?



```
% Is q(L) close to q(R)?

tol= 1e-4;  % tolerance
qL= L*L + b*L + c
qR= R*R + b*R + c
if (abs(qL-qR) < tol)
  disp('qleft and qright similar')
end
```

Name an important parameter and define it with a comment!

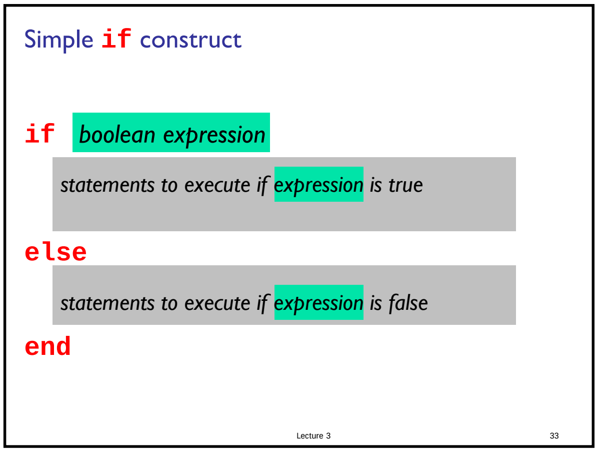


Do these two fragments do the same thing?

```
% given x, y
if  x>y
   disp('alpha')
else
   disp('beta')
end
```

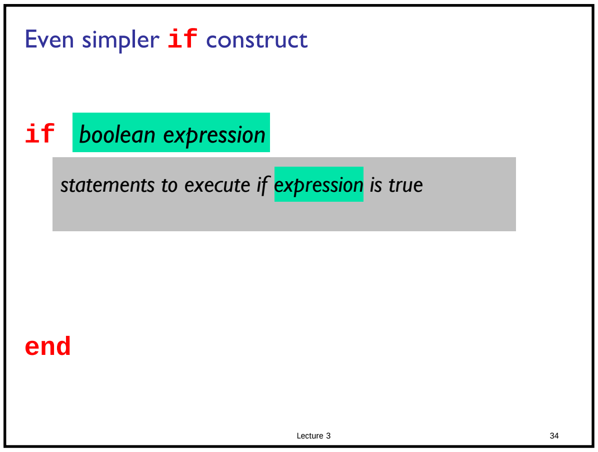
```
% given x, y
if  x>y
   disp('alpha')
end
if  y>=x
   disp('beta')
end
```

A: yes B: no



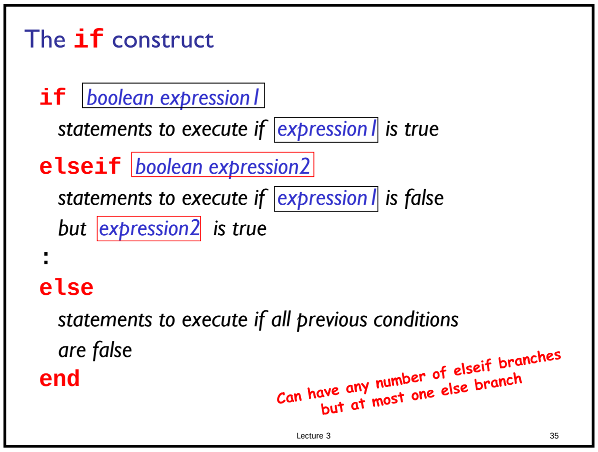
Simple if construct

```
if  boolean expression
    statements to execute if expression is true
else
    statements to execute if expression is false
end
```



Even simpler if construct

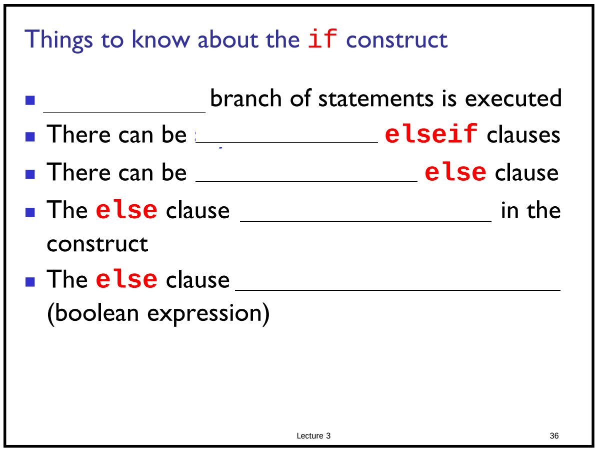
```
if  boolean expression
    statements to execute if expression is true


end
```



The if construct

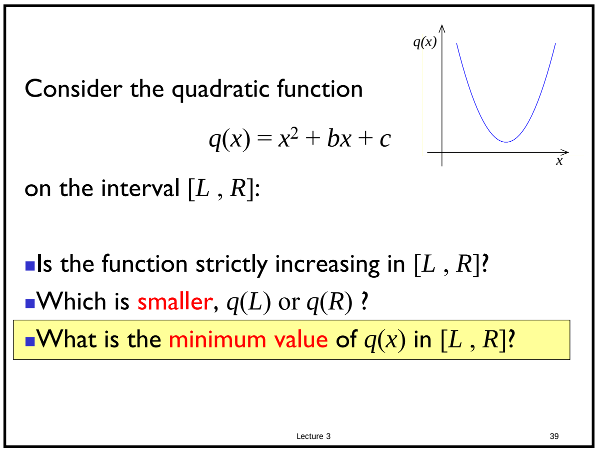
```
if  boolean expression1
    statements to execute if expression1 is true
elseif boolean expression2
    statements to execute if expression1 is false
    but expression2 is true
:
else
    statements to execute if all previous conditions
    are false
end
```

Can have any number of elseif branches but at most one else branch



Things to know about the if construct

- ________ branch of statements is executed
- There can be ________ elseif clauses
- There can be ________ else clause
- The else clause ________ in the construct
- The else clause ________ (boolean expression)

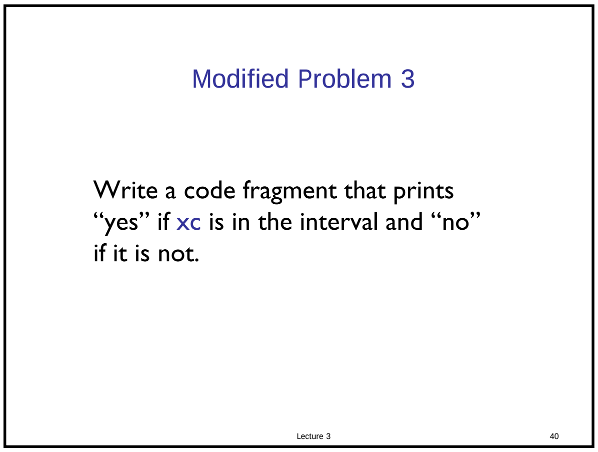


Consider the quadratic function

$$q(x) = x^2 + bx + c$$

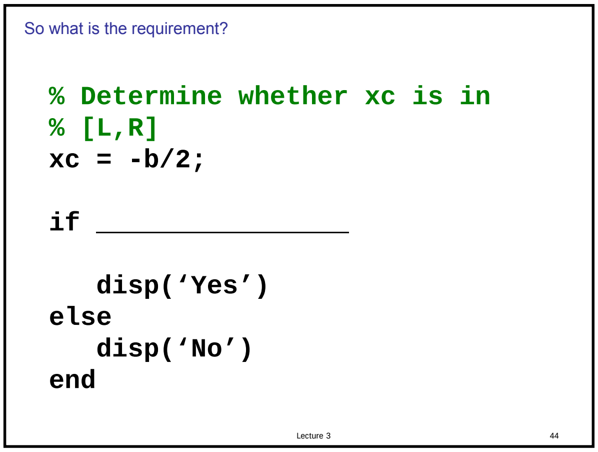
on the interval $[L\ ,\ R]$:

- Is the function strictly increasing in $[L\ ,\ R]$?
- Which is smaller, $q(L)$ or $q(R)$?
- What is the minimum value of $q(x)$ in $[L\ ,\ R]$?



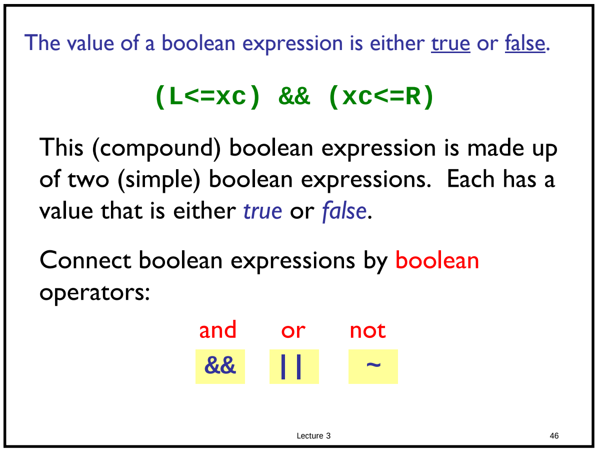
Modified Problem 3

Write a code fragment that prints “yes” if xc is in the interval and “no” if it is not.



So what is the requirement?

```
% Determine whether xc is in
% [L,R]
xc = -b/2;

if ________________

   disp(‘Yes’)
else
   disp(‘No’)
end
```



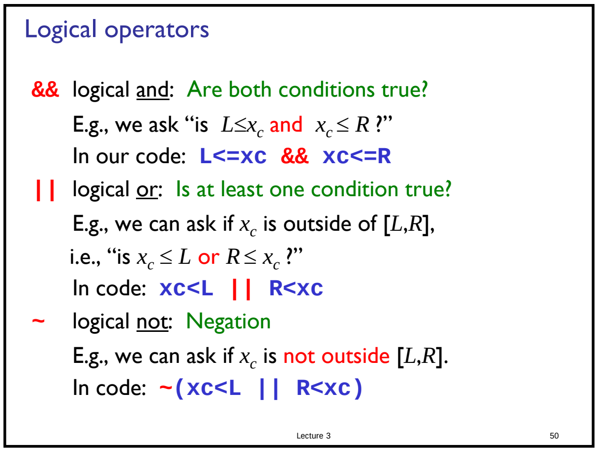
The value of a boolean expression is either true or false.

```
(L<=xc) && (xc<=R)
```

This (compound) boolean expression is made up of two (simple) boolean expressions. Each has a value that is either *true* or *false*.

Connect boolean expressions by boolean operators:

and	or	not
&&	||	~



Logical operators

&& logical and: Are both conditions true?

E.g., we ask “is $L \le x_c$ and $x_c \le R$?”

In our code: `L<=xc && xc<=R`

|| logical or: Is at least one condition true?

E.g., we can ask if x_c is outside of $[L,R]$,

i.e., “is $x_c \le L$ or $R \le x_c$?”

In code: `xc<L || R<xc`

~ logical not: Negation

E.g., we can ask if x_c is not outside $[L,R]$.

In code: `~(xc<L || R<xc)`