- 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:

- Project 1 (P1) due Thurs, 9/9, at 11pm
- Pay attention to Academic Integrity
- TAs: See any TA for help, not just your section instructor
- Consulting
  - Matlab consultants at ACCEL Green Rm (Engrg Library 2nd fl. computing facility)
  - 4-10pm Sunday to Thursday (except 9/12, 9/13)
- You need to attend the section in which you are enrolled
- Just added CS1112? Tell your discussion TA to add you in CS1112 CMS (and tell CS1110 to drop your from their CMS).
- Register your clicker with Cornell CIT (see course website)

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?

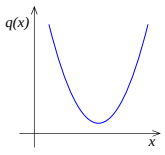
    x = y or y = x
- Script (program)
  - A sequence of statements saved in an m-file
- ; (semi-colon)
  - Suppresses printing of the result of assignment statement

- 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

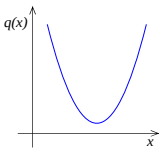
$$q(x) = x^2 + bx + c$$

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: $x = L$, $x = R$
  - $\{ x \mid q'(x) = 0 \}$



$$q(x) = x^2 + bx + c$$
$$q'(x) = 2x + b$$
$$q'(x_c) = 0 \Rightarrow x_c = -\frac{b}{2}$$

Problem 1

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

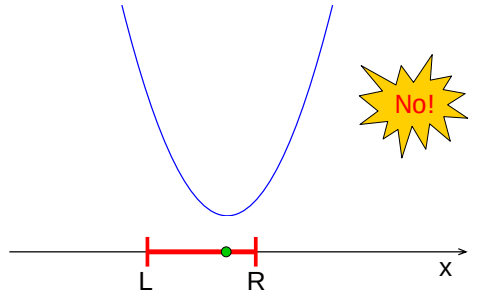

```
% Quadratic q(x) = x^2 + bx + c
b = input('Enter b: ');
c = input('Enter c: ');
L = input('Enter L: ');
R = input('Enter R: ');

% 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

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."

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

Algorithm v2

Calculate x_c

If distance $\overline{x_cL}$ is same as distance $\overline{x_cR}$

    print "qleft and qright are equal"

Otherwise, if $\overline{x_cL}$ is shorter than $\overline{x_cR}$

    print "qleft is smaller"

Otherwise

    print "qright is smaller"

```
% Which is smaller, q(L) or q(R)?

xc= -b/2;  % x at center
if (abs(xc-L) == abs(xc-R))
   disp('qleft and qright are equal')
elseif (abs(xc-L) < abs(xc-R))
   disp('qleft is smaller')
else
   disp('qright is smaller')
end
```


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

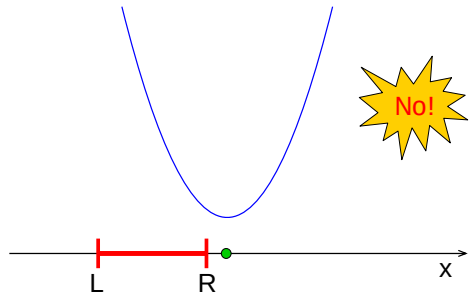
Things to know about the `if` construct

- _____ 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)

Modified Problem 3

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

$q(x) = x^2 + bx + c$ $x_c = -b/2$



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

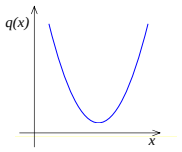
&& logical and: Are both conditions true?
E.g., we ask "is $L \le x_c$ and $x_c \le R$?"
In our code: `L<=xc && xc<=R`

|| 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)`

Consider the quadratic function

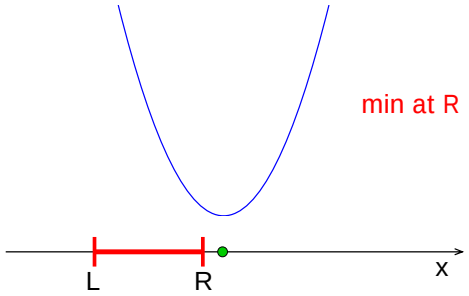
$$q(x) = x^2 + bx + c$$



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 $[L , R]$?

$q(x) = x^2 + bx + c$ $\quad$ $x_c = -b/2$



Start with pseudocode

If *xc* is between *L* and *R*

Min is at *xc*

Otherwise

Min is at one of the endpoints

We have decomposed the problem into three pieces! Can choose to work with any piece next: the if-else construct/condition, min at xc, or min at an endpoint

Set up structure first: if-else, condition

```
if  L<=xc && xc<=R
```

Then min is at *xc*

```
else
```

Min is at one of the endpoints

```
end
```

Now refine our solution-in-progress. I'll choose to work on the if-branch next

Refinement: filled in detail for task "min at xc"

```
if  L<=xc && xc<=R
   % min is at xc
   qMin= xc^2 + b*xc + c;

else
```

Min is at one of the endpoints

```
end
```

Continue with refining the solution… else-branch next

Refinement: detail for task "min at an endpoint"

```
if  L<=xc && xc<=R
   % min is at xc
   qMin= xc^2 + b*xc + c;
else
   % min is at one of the endpoints
   if  % xc left of bracket
      % min is at L
   else  % xc right of bracket
      % min is at R
   end
end
```

Continue with the refinement, i.e., replace comments with code

Final solution (given b,c,L,R,xc)

```
if  L<=xc && xc<=R
   % min is at xc
   qMin= xc^2 + b*xc + c;
else
   % min is at one of the endpoints
   if  xc < L
      qMin= L^2 + b*L + c;
   else
      qMin= R^2 + b*R + c;
   end
end
```

An if-statement can appear within a branch—just like any other kind of statement!

quadMin.m

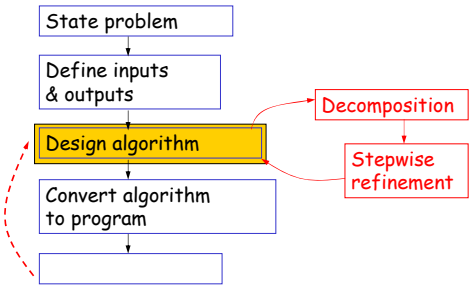
quadMinGraph.m

Notice that there are 3 alternatives→can use elseif!

```
if  L<=xc && xc<=R
  % min is at xc
   qMin= xc^2+b*xc+c;
else
  % min at one endpt
  if  xc < L
    qMin= L^2+b*L+c;
  else
    qMin= R^2+b*R+c;
  end
end
```

```
if  L<=xc && xc<=R
  % min is at xc
   qMin= xc^2+b*xc+c;
elseif xc < L
  qMin= L^2+b*L+c;
else
  qMin= R^2+b*R+c;
end
```


Top-Down Design



An algorithm is an idea. To use an algorithm you must choose a programming language and implement the algorithm.