



COMPUTER SKILLS

LESSON 7

Valeria Cardellini
cardellini@ing.uniroma2.it
A.Y. 2015/16

Objectives of this lesson

We'll discuss

- Code blocks
- Selection statements: useful when you may want to execute some parts of the code **under certain circumstances only**
 - **if** statement
 - **if-else** statement
 - Nested **if-else** statement
 - **switch** statement

Code block

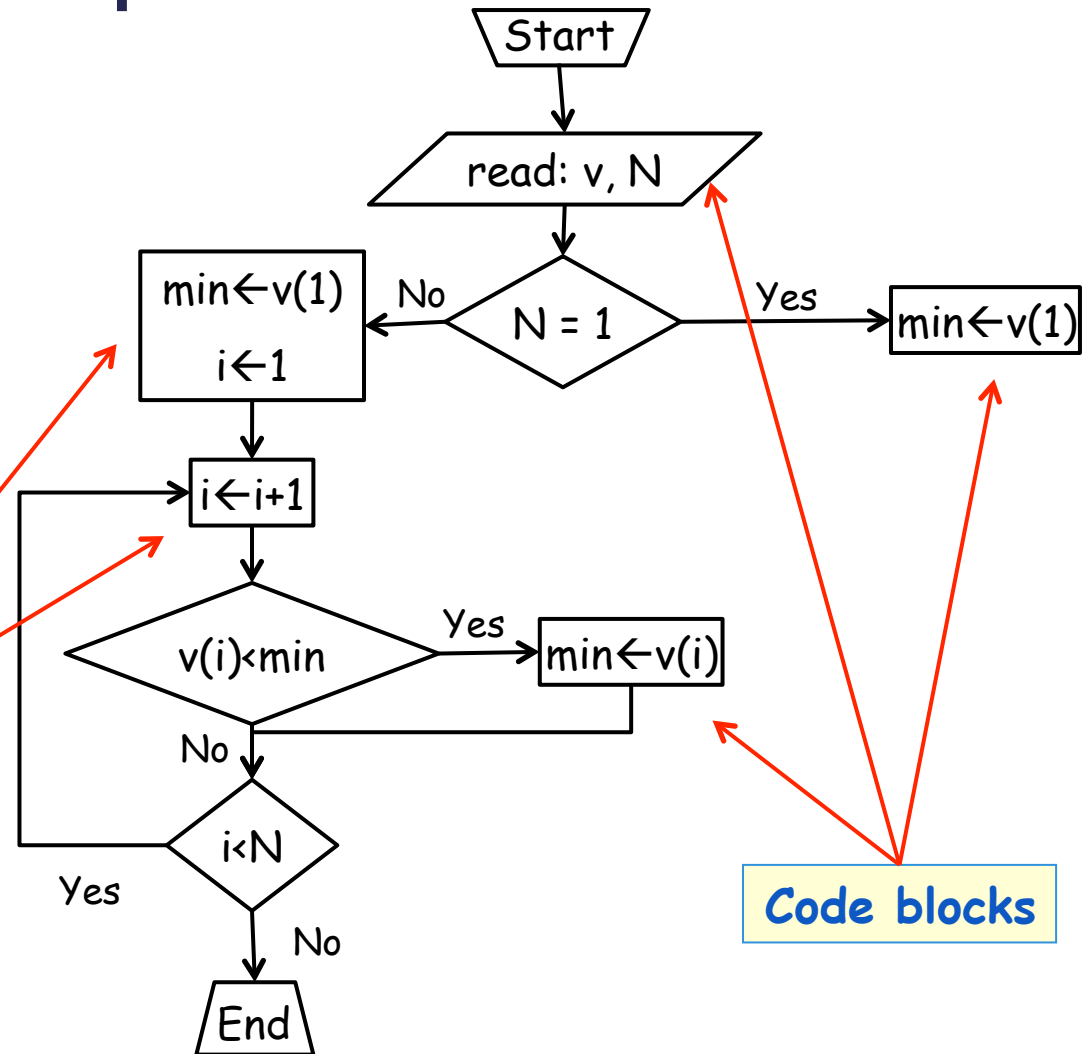
- A code block is a collection of zero or more MATLAB instructions identified for one of two reasons:
 1. you want to execute them only under certain circumstances, or
 2. you want to repeat them a given number of times
- Some languages identify code blocks by enclosing them within curly brackets, others by the level of indentation of the text
- MATLAB uses the occurrence of **key command words** in the text to define the extent of code blocks:
 - `if, switch, while, for, case, otherwise, else, elseif, end`
- **Key command words** are identified with blue coloring by the MATLAB text editor. They are not part of the code block, but they serve both
 - as instructions on what to do with the code block, and
 - as delimiters that define the extent of the code block

Code block: example

- A code block is a collection of zero or more MATLAB instructions identified for one of two reasons:

1. you want to execute them only under certain circumstances, or
2. You want to repeat them a given number of times

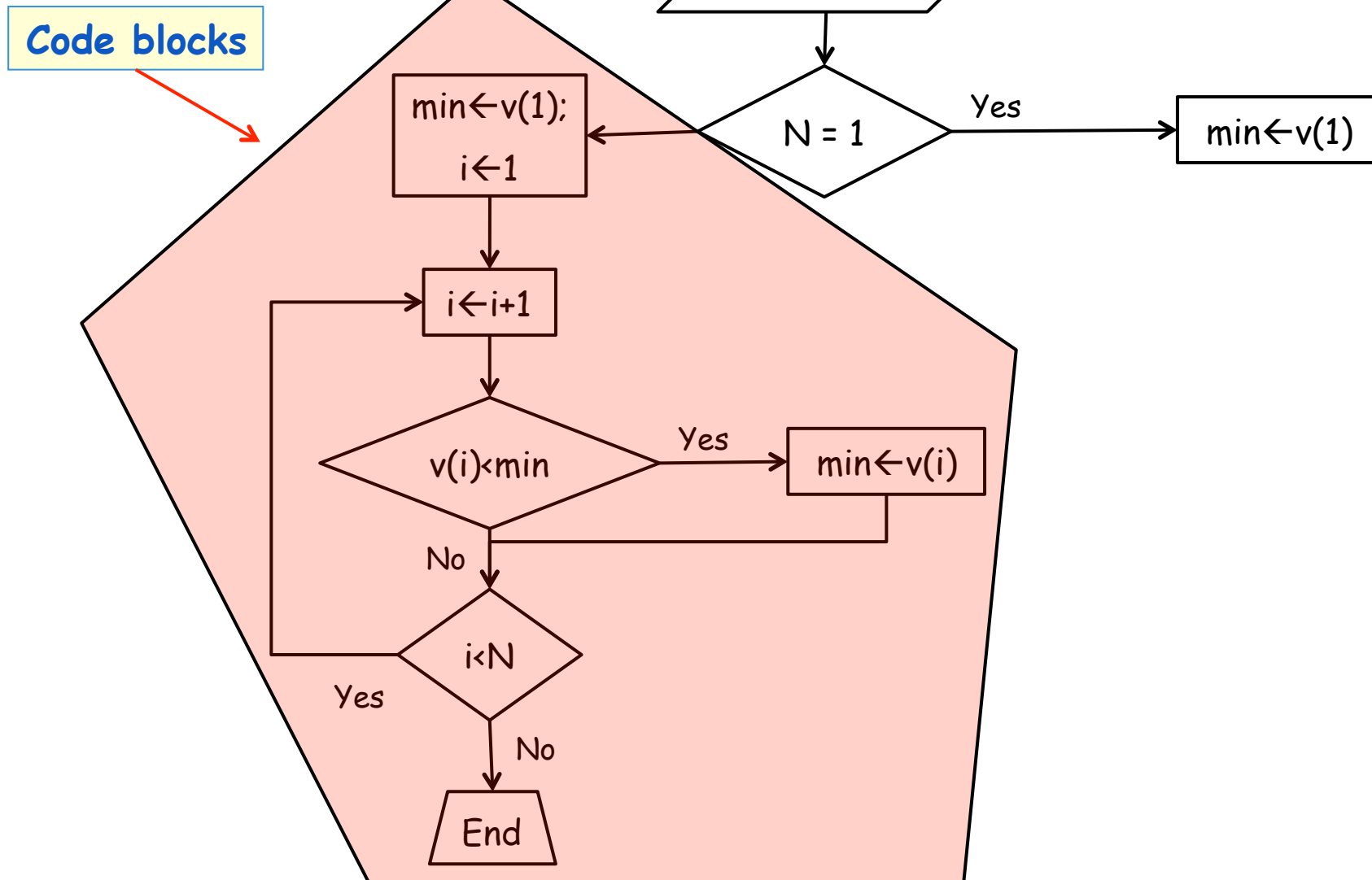
Code blocks



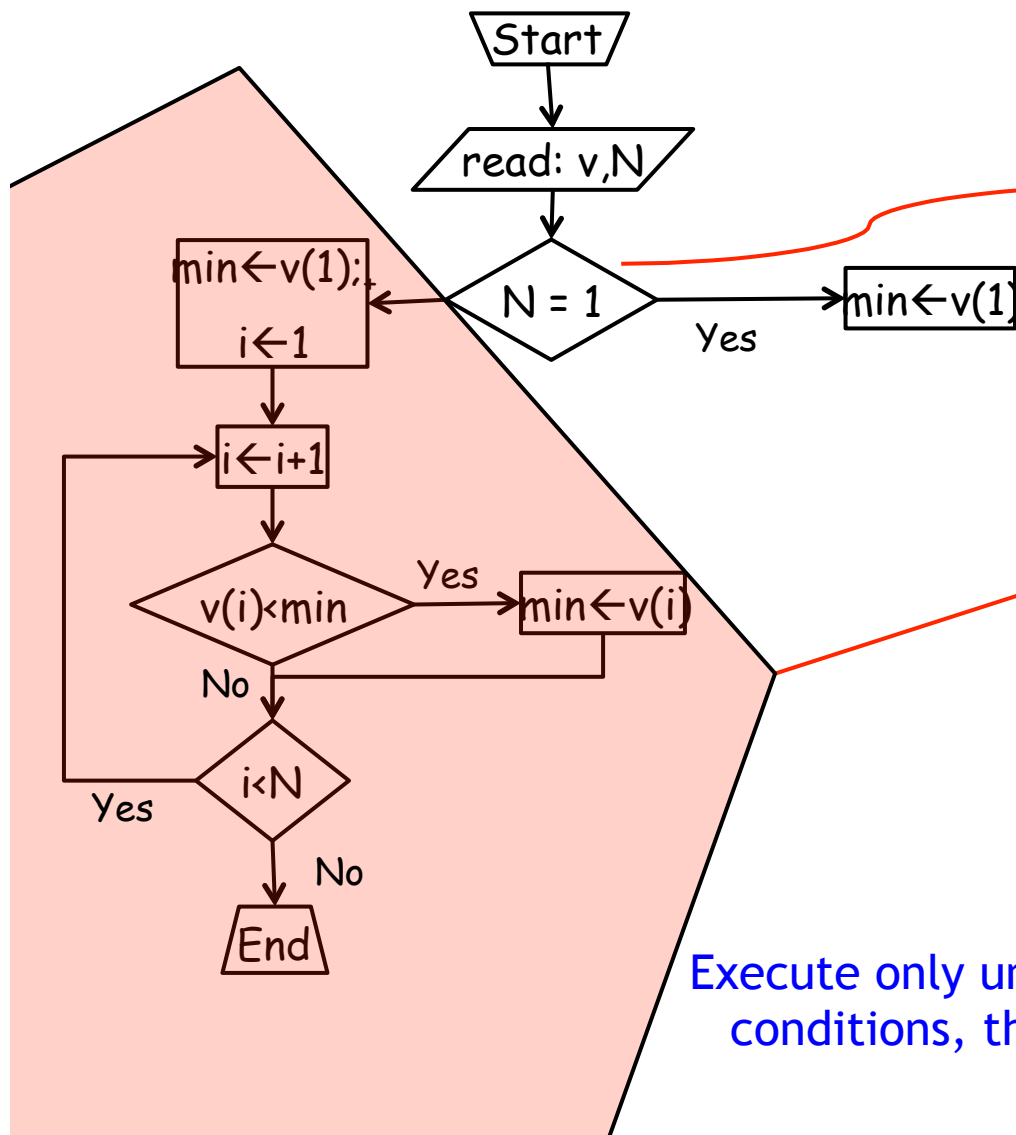
Code blocks

Flow diagram to find the minimum value of a vector

Code block: example



Code block: example



```
v=[4 5 2 19 5 7 8];
N=length(v);
```

```
if N==1
    mymin=v(1);
```

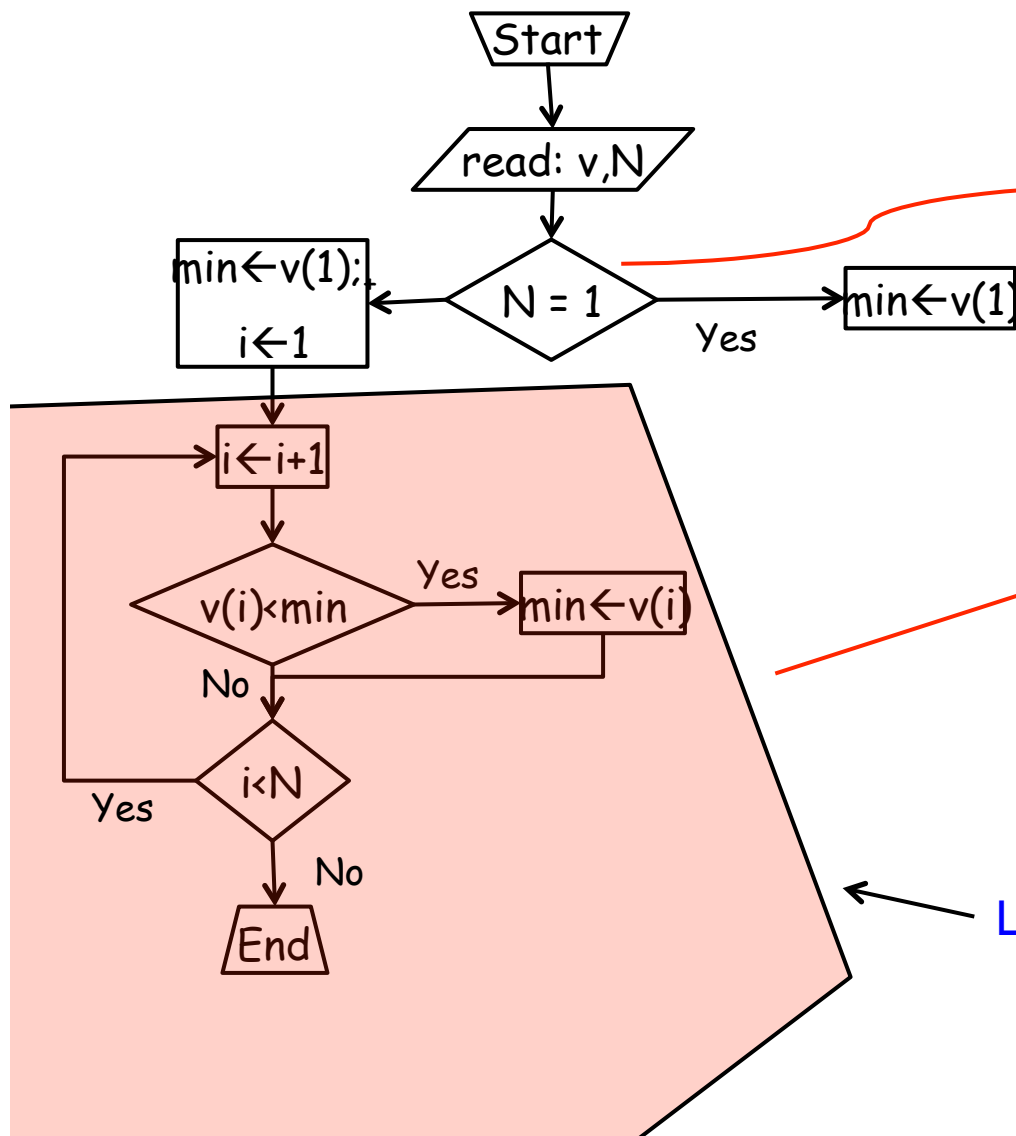
```
else
```

```
    mymin=v(1);
    i=2;
    while (i<=N)
        if v(i)<mymin
            mymin=v(i);
        end
        i=i+1;
    end
```

```
end
```

Execute only under certain conditions, that is $N > 1$

Code block: example



```
v=[4 5 2 19 5 7 8];  
N=length(v);
```

```
if N==1  
    mymin=v(1);
```

```
else  
    mymin=v(1);  
    i=2;
```

```
while (i<=N)  
    if v(i)<mymin  
        mymin=v(i);
```

```
    end  
    i=i+1;
```

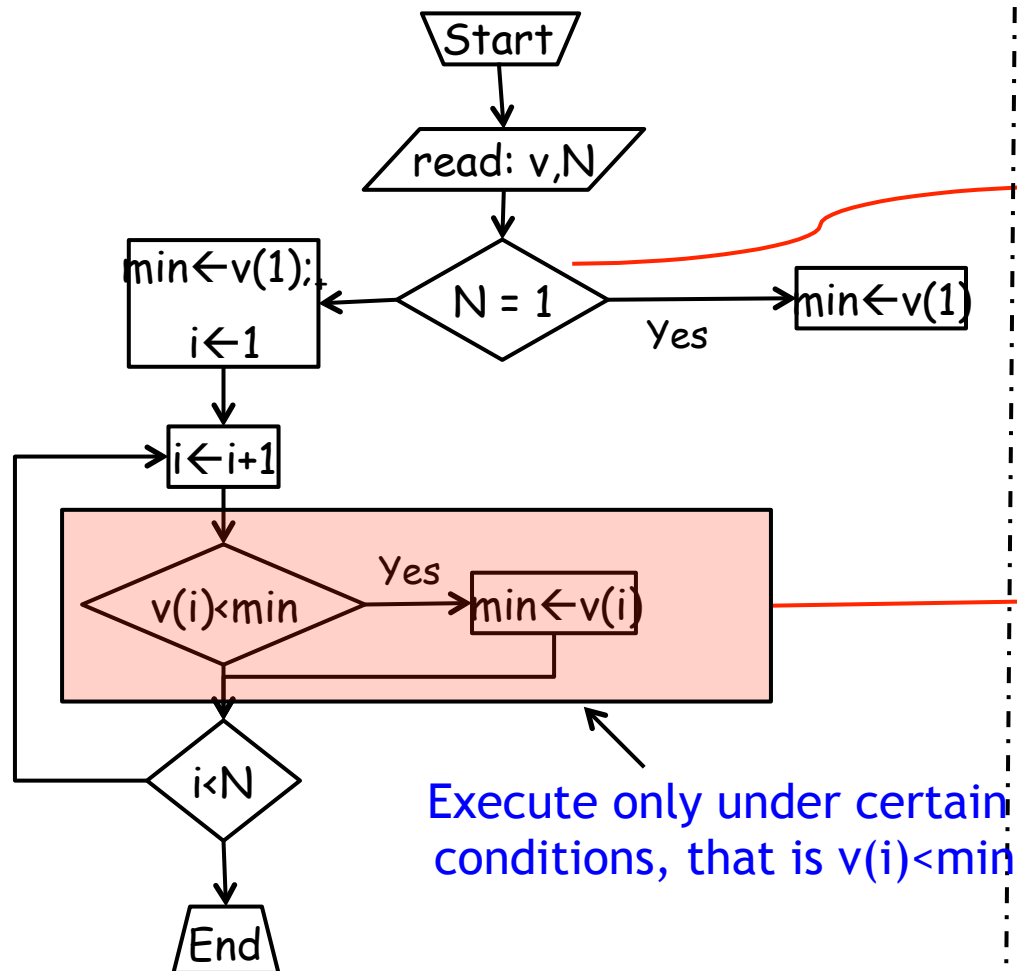
```
end
```

```
end
```

Loop repeated a certain number of times, that is $N-1$ times

Code blocks: example

min(v)



```
v=[4 5 2 19 5 7 8];
N=length(v);
```

```
if N==1
    mymin=v(1);
```

```
else
    mymin=v(1);
    i=2;
```

```
while (i<=N)
```

```
    if v(i)<mymin
        mymin=v(i);
```

```
    end
```

```
    i=i+1;
```

```
end
```

```
end
```


Code blocks: identification

- Some languages identify code blocks by enclosing them within curly brackets (e.g., C); others identify them by the level of indentation of the text (e.g., Python)

C

```
for(i=1;i<=m;i++)  
{  
  for(j=1;j<=n;j++)  
  {  
    x[i][j]=0;  
  }  
}
```

Python

```
x = 1  
while x < 5:  
    print ('Hi spam')  
    x = x + 1  
print ('done')
```

Code blocks: identification

- MATLAB uses the occurrence of **key command words** in the text to define the extent of code blocks:
 - **if, switch, while, for, case, otherwise, else, elseif, end**

MATLAB

```
for i=1:length(log3s_300);  
    for iterK=1:length(K)  
        start=1+(i-1)*(tau1)*W;  
        lambda=log3s_300(start:start+W*tau1-1);  
        Xmax=Tmax*max(lambda)/(Tmax*mu-1)*2;  
        filename=sprintf('output/%0.0fhK%sWl%0.0f',  
                           tau1, strK{iterK},start);  
        [x,TC,neval,XFlg, cost, avgViol] = optimalAllocation();  
    end  
end
```

Selection statements

- Selection statements allow to make choices as to whether statements are executed or not and how to choose between or among statements
- We will study:
 - **if** statement
 - **if** statement with **else** clause
 - **if** statement with **elseif** clause
 - **switch** statement

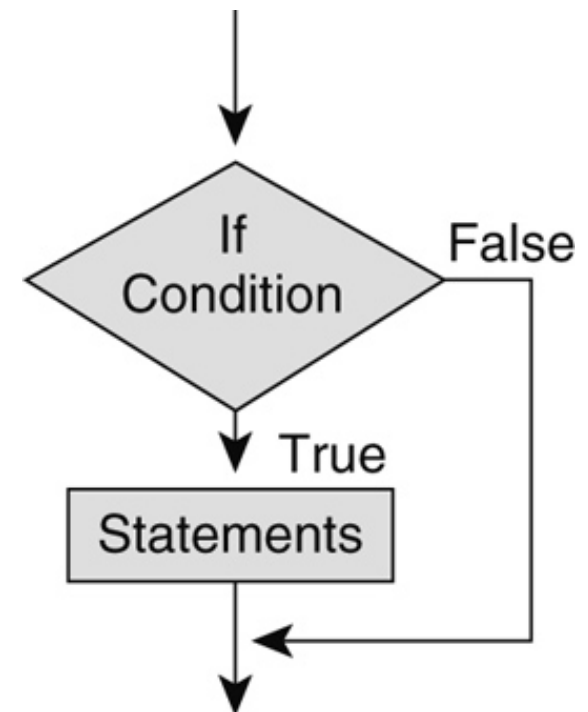
Conditional execution

- The basic conditional execution requires two elements:
 - A logical expression (condition)
 - A code block
- If the expression is true, the code block is executed
- Otherwise, execution is resumed at the instruction following the code block
- The general form of the **if statement** is

```
if condition
    action
end
```

Example:

```
if v(i) < mymin
    mymin = v(i);
end
```



Example of `if` statement

- A script that prompts the user for a number and prints the square root. If the user enters a negative number, the absolute value of the number will be used

See `sqrtifexampii.m`

```
num = input('Please enter a number: ');  
if num < 0  
    disp('Ok, the absolute value will be used');  
    num = abs(num);  
end  
fprintf('The sqrt of %.1f is %1f\n', num, sqrt(num));
```

A more complex condition

- Let us consider the problem of the top 10 “qualified” baseball players
- Let us define
 - yearsInLeague: vector containing the years in league
 - errorPerYears: vector containing the errors per year
 - plateApparance: vector containing the number of appearances per years

```
if yearsInLeague(i)>=5 && errorPerYears(i)<=10 && plateApparance(i)>100
```

```
    selectThePlayer(i)
```

```
end
```

Logical AND operator

A && B=true if A=true and B=true

A && B=false otherwise

if-else statement

- How to choose between two options?
- The general form of the **if-else statement** is

```
if condition
    action1
else
    action2
end
```

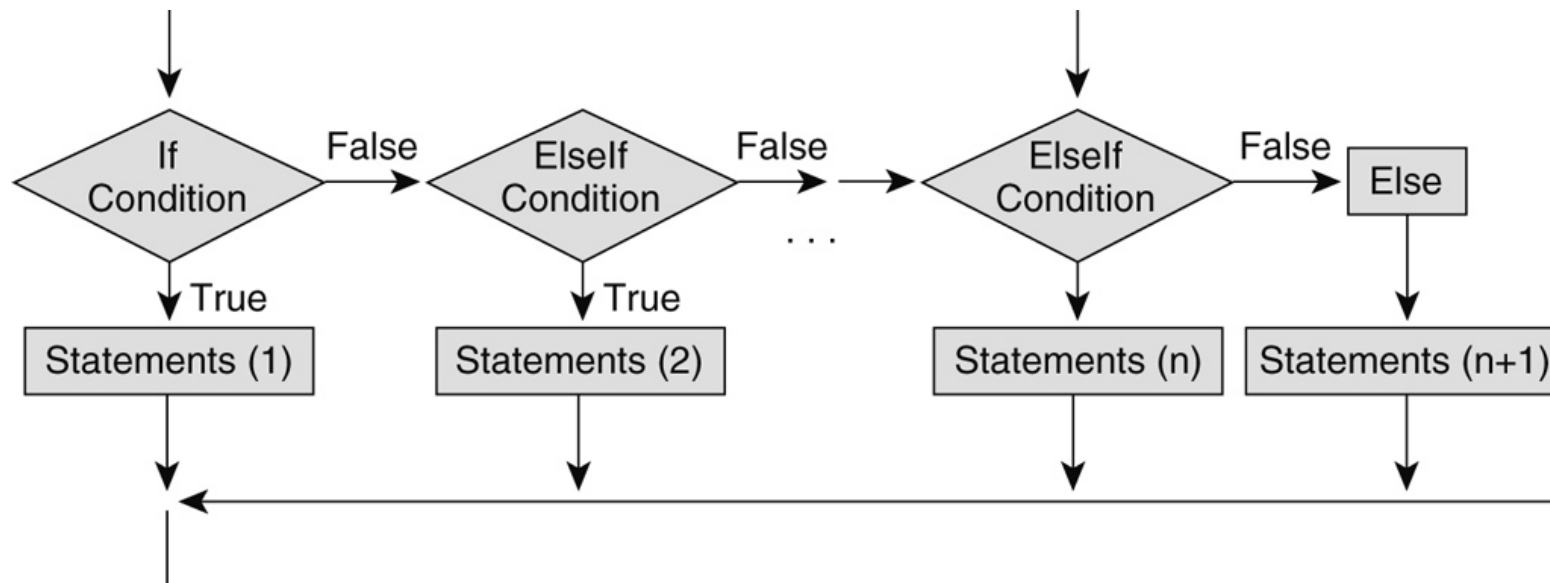
Example: print whether or not a random number in the range from 0 to 1 is less than 0.5

```
r = rand();
if r < 0.5
    disp('It was < 0.5');
else
    disp('It was >= 0.5');
end
```

Example of if-else statement

```
v=[4 5 2 19 5 7 8];  
N=length(v);  
if N==1  
    mymin=v(1);  
else  
    mymin=v(1);  
    i=2;  
    while (i<=N)  
        if v(i)<mymin  
            mymin=v(i);  
        end  
        i=i+1;  
    end  
end
```


Compound conditionals



- By introducing the **elseif clause**, we allow for the possibility of **choosing from multiple options**
- The same task can be accomplished using separate `if` statements or nested `if-else` statements, but using `elseif` is the simplest and most efficient way

if statement with elseif clause

- The general form for **if** statements with **elseif** clause is:

```
if <logical expression 1>
    <code block 1>
elseif <logical expression 2>
    <code block 2>
.
.
.
elseif <logical expression n>
    <code block n>
else
    <default code block>
end
```

Examples with `elseif` clause

```
if x < -1
    y = 1;
elseif x <= 2
    y = x^2;
else
    y = 4;
end
```

```
if temperature > 100
    disp('Too hot - equipment
        malfunctioning.')
elseif temperature > 90
    disp('Normal operating range.')
elseif temperature > 50
    disp('Below desired operating
        range.')
else
    disp('Too cold - turn off
        equipment.')
end
```

Choosing from multiple options

- Choosing from multiple options can be also accomplished using separate if statements or nested if-else statements, but using **elseif** is the simplest and most efficient way

Using elseif

```
if x < -1
    y = 1;
elseif x <= 2
    y = x^2;
else
    y = 4;
end
```

Using separate if

```
if x < -1
    y = 1;
end
if x >= -1 && x <= 2
    y = x^2;
end
if x > 2
    y = 4;
end
```

Using nested if-else

```
if x < -1
    y = 1;
else
    if x <= 2
        y = x^2;
    else
        y = 4;
    end
end
```

More examples

```
%% if statement example
day = input('enter a day(1-7): ');
if day == 7      % Saturday
    state = 'weekend';
elseif day == 1  % Sunday
    state = 'weekend';
else
    state = 'weekday';
end
fprintf('state=%s\n', state);
```

```
%% another example with if
grade = input('what grade? ');
if grade >= 90
    letter = 'A';
elseif grade >= 80
    letter = 'B';
elseif grade >= 70
    letter = 'C';
elseif grade >= 60
    letter = 'D';
else
    letter = 'F';
end
fprintf('letter=%c\n', letter);
```

- A code block can be stored as a MATLAB script and then executed

And more if...elseif...else...end

```
% Display text indicating whether x is a  
% scalar, vector, or matrix  
[m,n] = size(x);  
if m==n && m==1  
    disp('Argument is a scalar');  
elseif m==1 || n==1  
    disp('Argument is a vector');  
else  
    disp('Argument is a matrix');  
end
```

Logical OR operator

A || B=true if at least A=true or B=true

A || B = false otherwise

General observations

- A logical expression is any statement that returns a logical result
- If that result is a logical vector v , the if statement behaves as

if all(v)

```
% if statement with a logical vector
A = [true true false]
if A
    % will not execute
end
A(3) = true;
if A
    % will execute
end
```

- While indentation has no effect on the logical flow, it helps to clarify the logical flow
- MATLAB editor automatically creates suitable indentation as you type

switch statement

- The general form for the switch statement is:

```
switch <expression>  
  case <case specification 1>  
    <code block 1>  
  case <case specification 2>  
    <code block 2>  
    .  
    .  
  case <case specification n>  
    <code block n>  
  otherwise  
    <default code block>  
end
```


General observations

- The switch statement looks for the expression to have an exact match to one of the cases
- One case specification may have multiple values enclosed in curly brackets {...}
- The default case catches any values of the expression other than the specified cases
- The default case should trap bad expression values

```
%Example of switch statement
month = input('enter a month (1-12): ');
year = input('enter the year: ');
switch month
    case {4, 6, 9, 11}
        % Sept, Apr, June, Nov
        days = 30;
    case 2 % Feb
        if leapyear(year)
            %true if leap year
            days = 29;
        else
            days = 28;
        end
    case {1, 3, 5, 7, 8, 10, 12}
        % other months
        days = 31;
    otherwise
        error('bad month index')
end
days
```

switch case vs. if elseif

```
%Same example with if elseif
if month==4 || month==6 ||
month==9 || month==11
    % Sept, Apr, June, Nov
    days = 30;
elseif month==2 && leapyear(year)
    % Feb and leap year
    days = 29;
elseif month==2 && ~leapyear(year)
    % Feb and no leap year
    days = 28;
elseif month==1 || month==3 ||
month==5 || month==7 ||
month==8 || month==10 ||
month==12
    % other months
    days = 31;
else
    error('bad month index')
end
```

```
%Same example of previous slide
switch month
    case {4, 6, 9, 11}
        % Sept, Apr, June, Nov
        days = 30;
    case 2 % Feb
        if leapyear(year)
            %true if leap year
            days = 29;
        else
            days = 28;
        end
    case {1, 3, 5, 7, 8, 10, 12}
        % other months
        days = 31;
    otherwise
        error('bad month index')
end
```

See for a useful video:

blogs.mathworks.com/pick/2008/01/02/matlab-basics-switch-case-vs-if-elseif/

if-else vs. switch

- Problem statement
 - If interval is less than 1, set the value of xinc equal to interval/10; otherwise, set the value of xinc equal to 0.1

if-else

```
if interval < 1
    xinc = interval/10;
else
    xinc = 0.1;
end
```

switch

```
switch interval < 1
    case 1
        xinc = interval/10;
    case 0
        xinc = 0.1;
end
```

if-else versus switch: hint

%if-else

```
if interval < 1
    xinc = interval/10;
else
    xinc = 0.1;
end
```



%if-else

```
a=interval < 1;
if a
    xinc = interval/10;
else
    xinc = 0.1;
end
```

%switch

```
switch interval < 1
    case 1
        xinc = interval/10;
    case 0
        xinc = 0.1;
end
```



%switch

```
a=interval < 1;
switch a
    case 1
        xinc = interval/10;
    case 0
        xinc = 0.1;
end
```