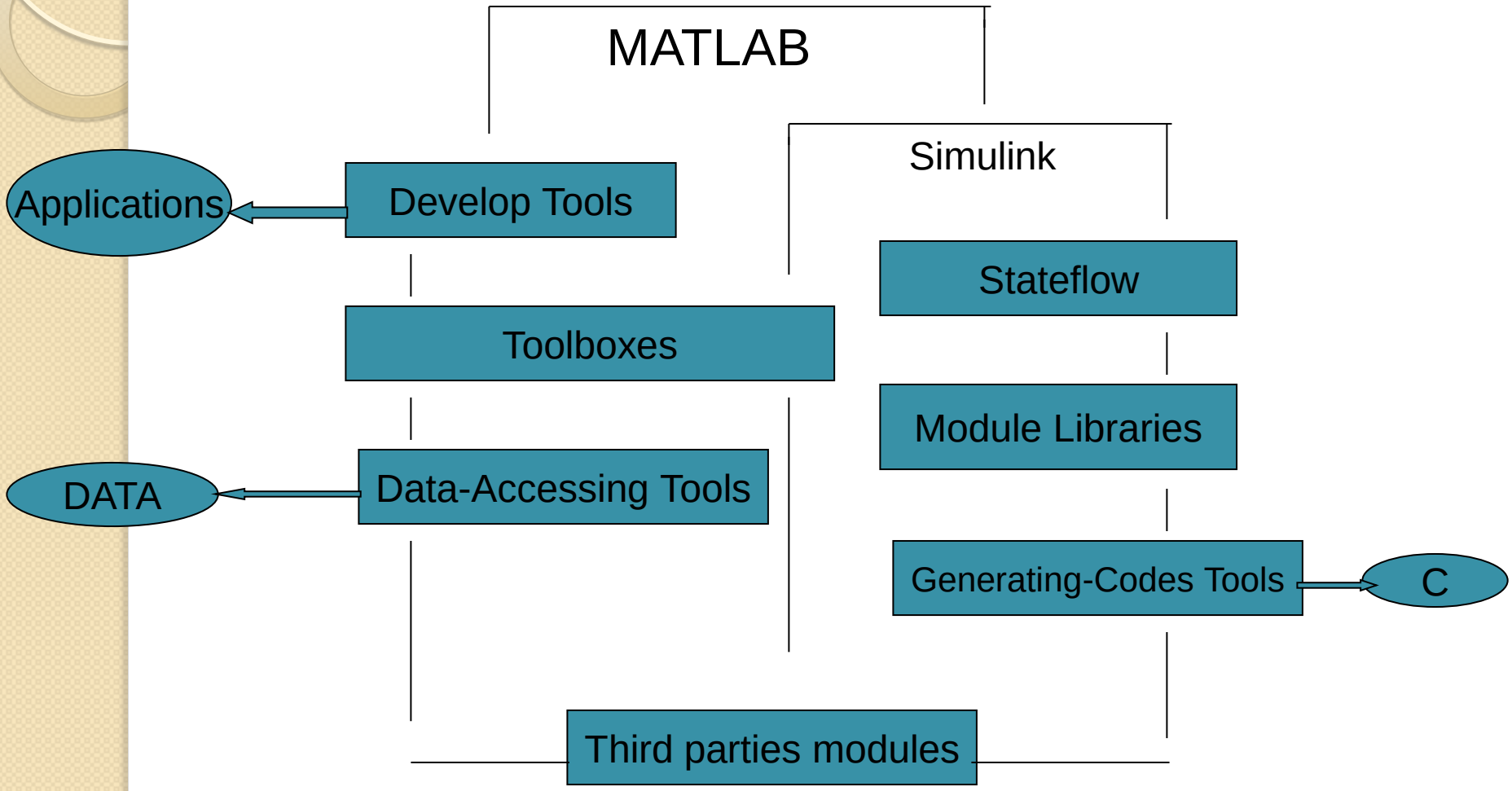
CE300 MATLAB PROGRAMMING

By Soner Seçkiner

OBJECTIVES

- OVERVIEW
- RELATIONAL AND LOGICAL OPERATORS
- CONDITIONALS
- FLOW CONTROLS
- ERRORS AND DEBUGGING
- SOME EXAMPLES

Overview


MATLAB
Simulink
Applications
Develop Tools
Stateflow
Toolboxes
Module Libraries
DATA
Data-Accessing Tools
Generating-Codes Tools
C
Third parties modules

Overview

- **Development Environment**

  MATLAB desktop and Command Window, a command history, an editor and debugger, and browsers for viewing help, the workspace, files, and the search path.
- **The MATLAB Mathematical Function Library**

  A vast collection of computational algorithms ranging from elementary functions to more sophisticated functions
- **The MATLAB Language**

  A high-level matrix/array language with control flow statements, functions, data structures, input/output, and object-oriented programming features
- **Graphics**

  Display vectors and matrices as graphs

  two-dimensional and three-dimensional data visualization

  image processing

  animation and presentation graphics
- **The MATLAB Application Program Interface (API)**

  Interchange the C/Fortran codes with Matlab codes

RELATIONAL AND LOGICAL OPERATORS

- Relational operators
  - < ,> , <=, >=, ==, ~=
  - isnan(), isinf()
- Logical Operators
  - &&, || , ~, xor

A	B	~A	A || B	A && B	xor(A,B)
0	0	1	0	0	0
0	1	1	1	0	1
1	0	0	1	0	1
1	1	0	1	1	0

RELATIONAL AND LOGICAL OPERATORS – Hierarchy of Operators

- Arithmetic operations (paranthesis (innermost towards outermost), exponentials,* and / from left to right,+ and – from left to right)
- Relational operations from left to right
- ~
- & from left to right
- | from left to right

CONDITIONALS

- **Simple if Statement**

```
if logical expression
        commands
End
```

```
if d < 50
   count = count + 1;
   disp(d);
end
```

CONDITIONALS

- **Nested if Statements**

```
if d < 50
    count = count + 1;
    disp(d);
    if b > d
        b = 0;
    end
end
```

CONDITIONALS

- Else **and** elseif **Clauses**

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

CONDITIONALS

- Switch clause

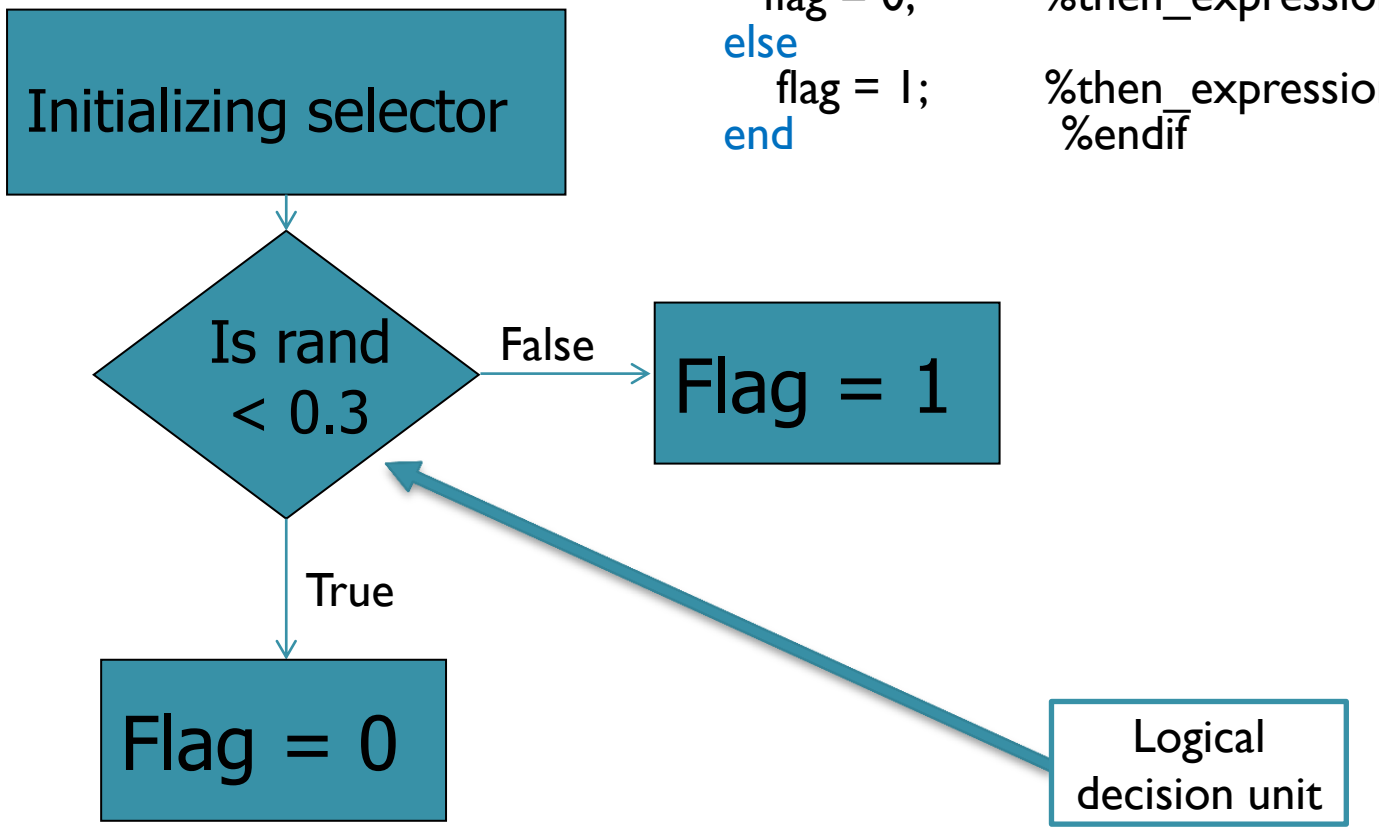
```
switch expression
    case test expression 1
    commands
    case {test expression 2, test expression 3}
    commands
    .
    .
    .
    otherwise
    commands
end
```

```
x = 6.1;
units = 'ft';
% convert x to meters
switch units
        case {'inch','in'}
        y = x*0.0254;
        case{'feet','ft'}
        y = x*0.3048;
        case{'meter','m'}
        y = x;
        case{'centimeter','cm'}
        y = x/100;
        case{'millimeter','mm'}
        y = x/1000;
        otherwise
        disp(['Unknown units: ' units])
        y = NaN;
end
```

FLOW CONTROLS

- Flow control

```
if rand < 0.3        %Condition
   flag = 0;         %then_expression_1
else
    flag = 1;        %then_expression_2
end                   %endif
```



FLOW CONTROLS

Multiple-ways:

```
if A > B
   'greater‘
elseif A < B
   'less‘
elseif A == B
   'equal‘
else
   error('Unexpected situation')
end
```

```
switch (grade)
  case 0
       ‘Excellent’
  case 1
       ‘Good’
  case 2
       ‘Average’
  otherwise
       ‘Failed’
end
```

FLOW CONTROLS

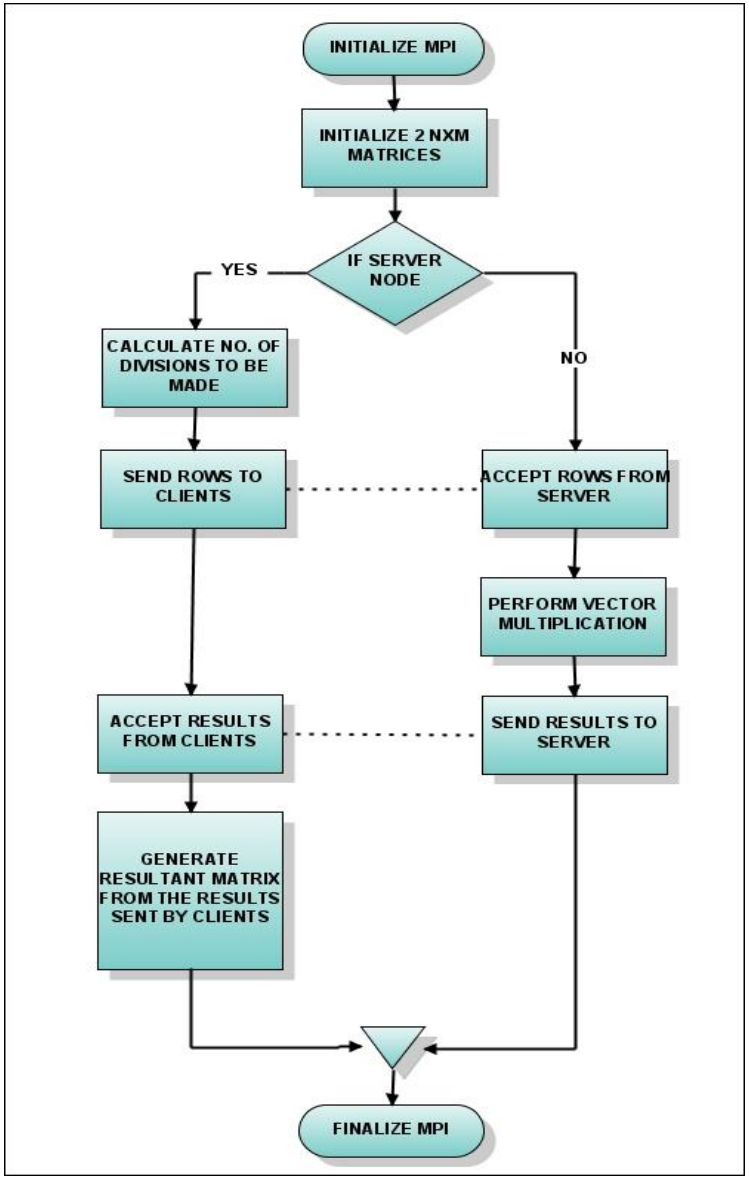
- if, else, elseif executes a group of statements based on some logical condition.
- Switch, case, otherwise executes different group of statements depending on the value of some logical condition.
- While executes a group of statements an indefinite number of times, based on some logical condition.

FLOW CONTROLS

- For executes a group of statements a fixed number of times.
- Continue passes control to the next iteration of a for or while loop, skipping any remaining statements in the body of the loop
- Break terminates execution of a for or while loop.
- Return causes execution to return the invoking function.
- All flow constructs use end to indicate the end of the flow control block.

FLOW CONTROLS

- Example Flow Diagram MPI API



FLOW CONTROLS

- For Loop

>>Generate a matrix

```
for i = 1:1:5
        for j = 1:6
            H(i,j) = 1/(i+j);
        end
end
```

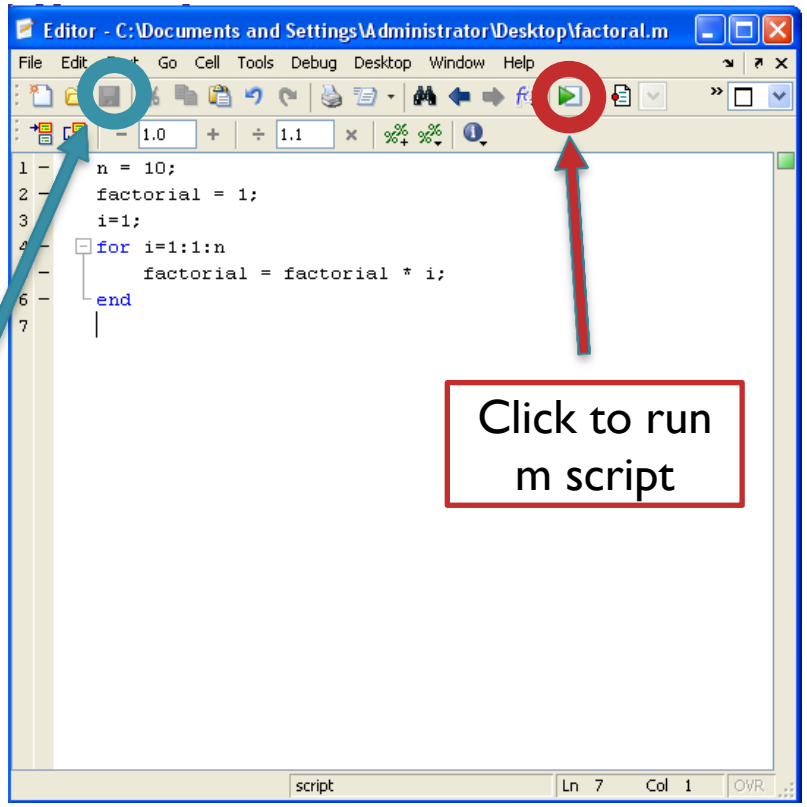
FLOW CONTROLS

- While Loop

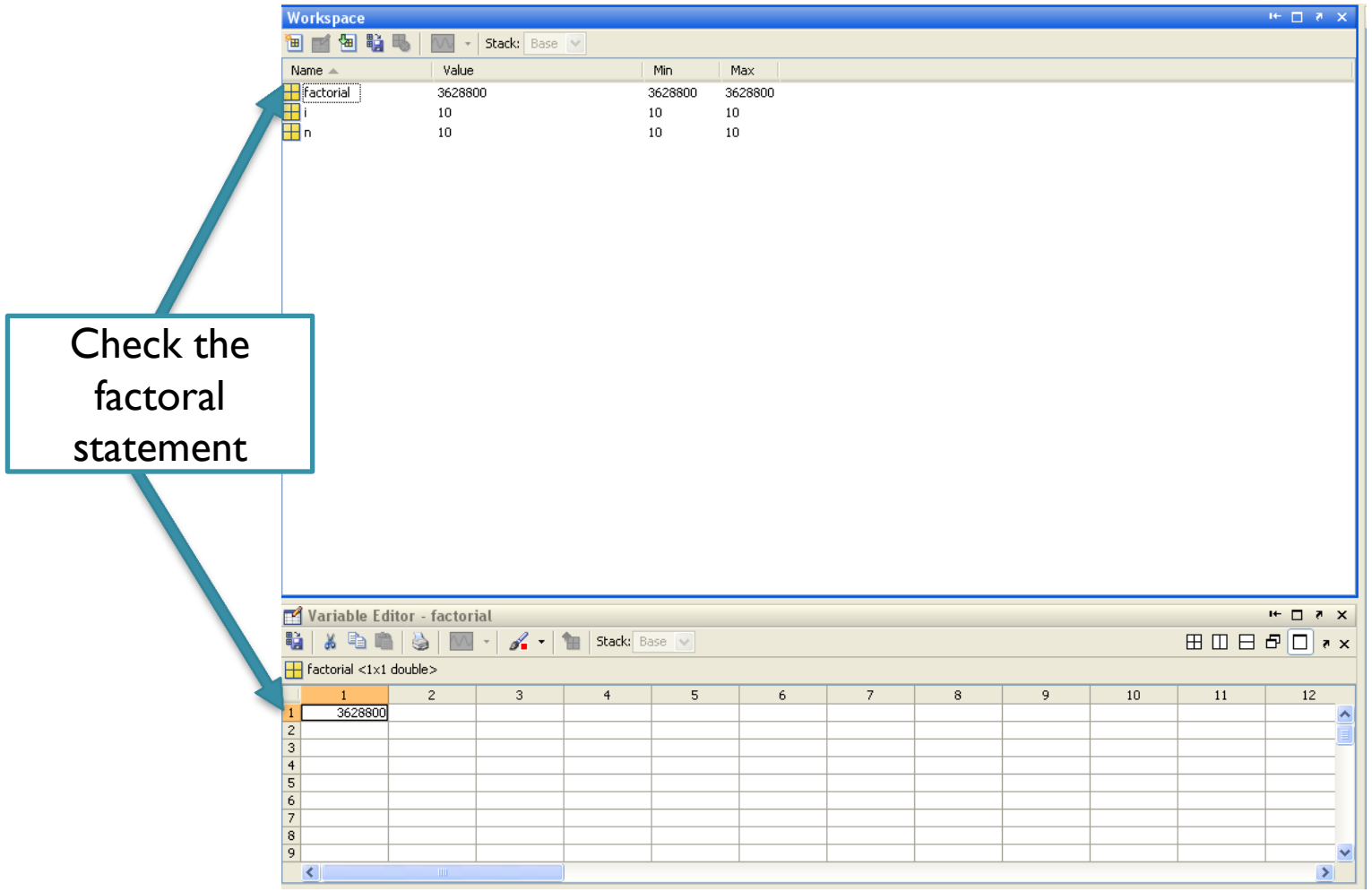
```
>>%Obtain the result
a = 0; fa = -Inf;
b = 3; fb = Inf;
while b-a > eps*b
   x = (a+b)/2;
   fx = x^3-2*x-5;
   if sign(fx) == sign(fa)
      a = x; fa = fx;
       break;
   else
      b = x; fb = fx;
      continue;
   end
    fa = x^3-2*x-5;
end
```

Errors and Debugging - M File

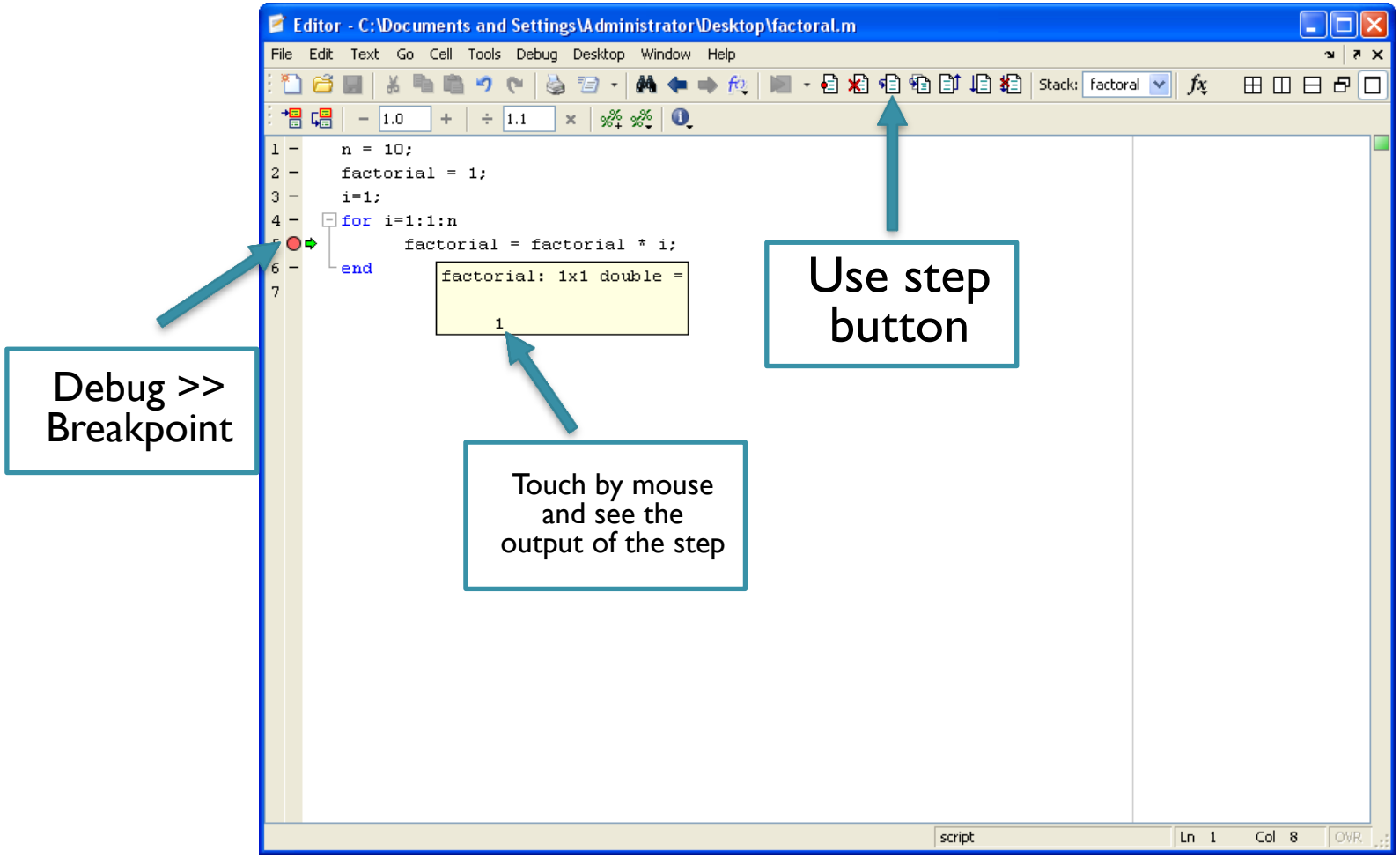
- n = 10;
- factorial = 1; % comment
- i=1;
- for i=1:1:n
- 	factorial = factorial * i;
- end



Errors and Debugging



Errors and Debugging



Errors and Debugging – Common Errors

- Missing parenthesis
  - 4*(2+5
- Missing operator
  - 4(2+5)
- Misspelled variable name
  - 2x = 4*(2+5)
- Run-time and Logic Errors
  - Be carefull about
    - NaN (not a number),
    - Inf (infinity)
    - Empty array
  - Some advises
    - Execute simple parts alone for testing
    - Use tab for the logical expressions and loops
    - Use comments for other programmers
    - Test your program by using debug menu
    - Construct flow diagram for your algorithm

Some Examples

- % Matrix multiplication example using inner products

```
for i=1:n,
    for j=1:n
        C(i,j) = A(i,:)*B(:,j);
     end
end
```

Some Examples

- Simple average operation

```
% initialize - prepare to read 1st datum
i = 1;
% read and count data values
data = input('Enter datum ("Enter" to stop): ');
while ~isempty(data)     %data?
   y(i) = data;          % - yes: store
   i = i+1;              %       count
   data = input('Enter datum ("Enter" to stop): ');
end

% no more data - compute average
sumY = sum(y);           % compute sum
[dummy, n] = size(y);    % determine # values = # columns
averageY = sumY/n;

% print result
disp(['the average of the ', num2str(n) , ' values is ',  num2str(averageY)])
```

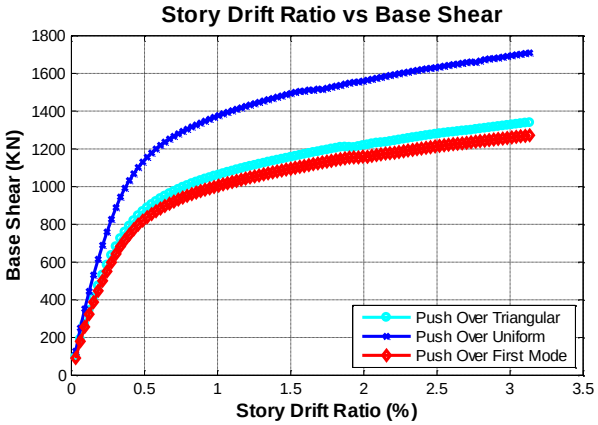
Some Examples

- Cost estimation example

```
% initialize parameters
baseRate = 10;
baseWeight = 2;
maxStandardWeight = 70;
additional = 3.75;
overweight = 10;

% read and validate weight
weight = input('Enter package weight(lb.): ');
if isempty(weight)
   disp('you MUST enter a weight!')
elseif weight > 100
   disp(['weight of ', num2str(weight), ' exceeds 100 pounds - please see Federal Express.'])
% compute shipping cost
else
   % standard package
   cost = baseRate + ceil(weight-baseWeight)*additional;
   % overweight charge
   if (weight > maxStandardWeight)
      cost = cost + overweight;
   end
end
% print results
disp(['total shipping cost for ', num2str(weight), ' pounds is $', num2str(cost)]);
```

Some Examples



- Simple plot example

```
figure('Position',size);
for k=1:1:analysis_cases
    clr= color(mod(k*7,6)+1);
    typ= type(mod(k-1,6)+1);
    ptype=cell2mat(strcat(clr,typ));
    plot (ISD(data_start:rows,1),wallshearpercent(:,k),ptype,'LineWidth',2,'MarkerSize',5);
    hold on;
end

title('Story Drift Ratio vs Wall Percentage','FontSize',fs_title,'FontWeight','Bold');
xlabel('Story Drift Ratio (%)','FontSize',fs_axes,'FontWeight','Bold');
ylabel('Wall percentage (%)','FontSize',fs_axes,'FontWeight','Bold');
h = legend('Push Over Triangular', 'Push Over Uniform', 'Push Over First Mode', 1);
set(h,'Interpreter','none','FontSize',fs_legend);
grid on;
hold off;
figurename =strcat('wallpercent_',building);
print(figurename,'-r600', '-dmeta');
```

Any Questions?