1- Control Statement

In computer science, control flow is the order in which individual statements, instructions or function calls of an imperative program are executed or evaluated.

Within an imperative programming language, a control flow statement is a statement whose execution results in a choice being made as to which of two or more paths should be followed.

The kinds of control flow statements supported by different languages vary, but can be categorized by their effect:

- Continuation at a different statement.
- Executing a set of statements only if some condition is met.
- Executing a set of statements zero or more times, until some condition is met
- Executing a set of distant statements, after which the flow of control usually returns.
- Stopping the program, preventing any further execution.

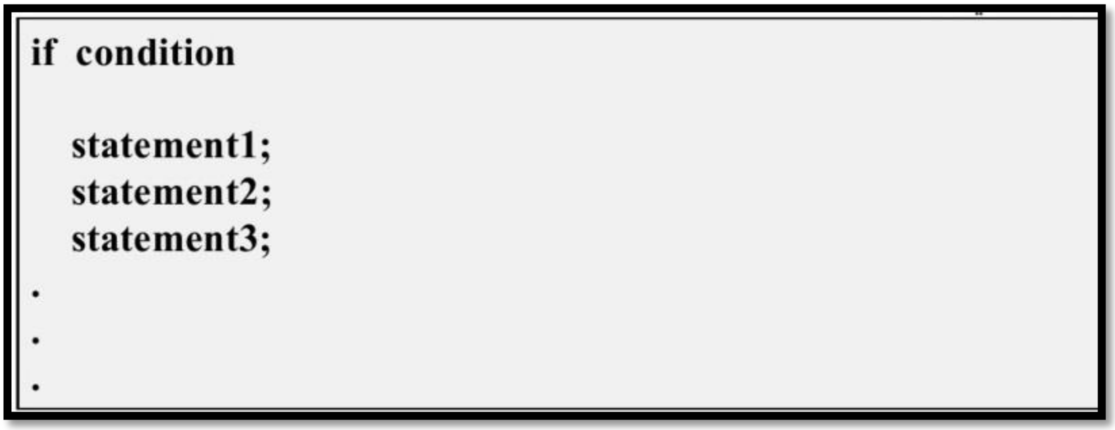
Control Statements in Matlab can be classified into four types:

1- Conditional Statements:
   a- IF Statement.
   b- Switch Statement.
2- Looping Statements:
   a- For Loops Statement.
   b- While Loops Statement.
3- Jumping Statements:
   a- Break Statement.
   b- Continue Statement.
4- Exception Statement.

a- Try-catch statement.

1.1 If Statement

An if statement is a selection statement that allows more than one possible flow of control.

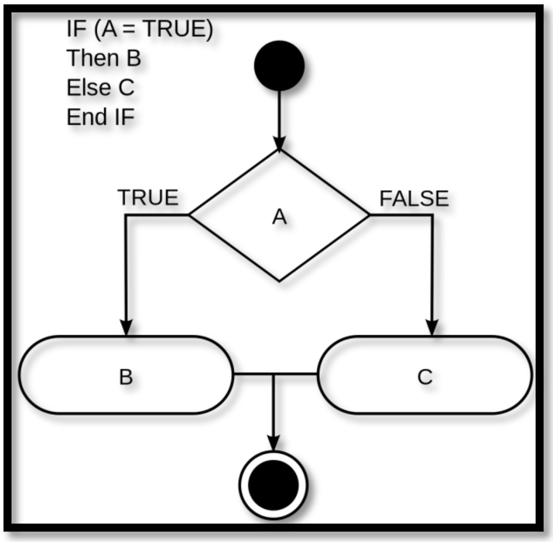


```
if  condition

   statement1;
   statement2;
   statement3;
.
.
.
```

1.2 If…Else Statement

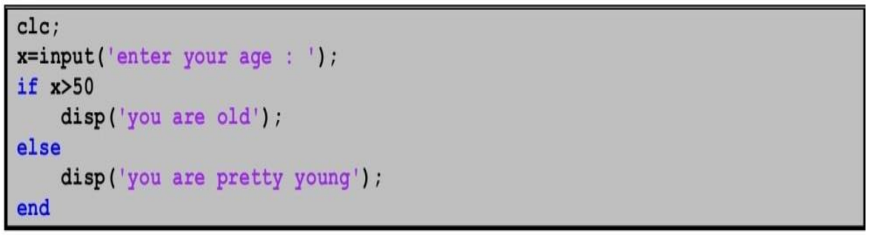
If the expression evaluates to a non-zero quantity, the following statement or block is executed, and the block or statement following the **else** keyword is not executed. Otherwise, control flow jumps to the block or statement following the **else** keyword, or simply to after the entire if statement.



```
if  condition

   statement1;
   statement2;
   ……
Else
statement3;
statement4;
……
end
```

Example 1: write a program that reads the number that we are entered and then comparing the entrance results.



```
clc;
x=input('enter your age : ');
if x>50
    disp('you are old');
else
    disp('you are pretty young');
end
```

1.3 If…Elseif Statement

```
if  condition1
   statement1;
……..
elseif  condition2
   statement2;
   ……
elseif  condition(n-1)

Statement(n-1);
```

Very often when you write code, you want to perform different actions for different decisions. You can use conditional statements in your code to do this.

- **if statement** - executes some code only if a specified condition is true.
- **if...else statement** - executes some code if a condition is true and another code if the condition is false.
- **if...elseif....else statement** - specifies a new condition to test, if the first condition is false.

Example 2: write a program to calculate the summations of 5 degrees and then find the average for these grades and check it:

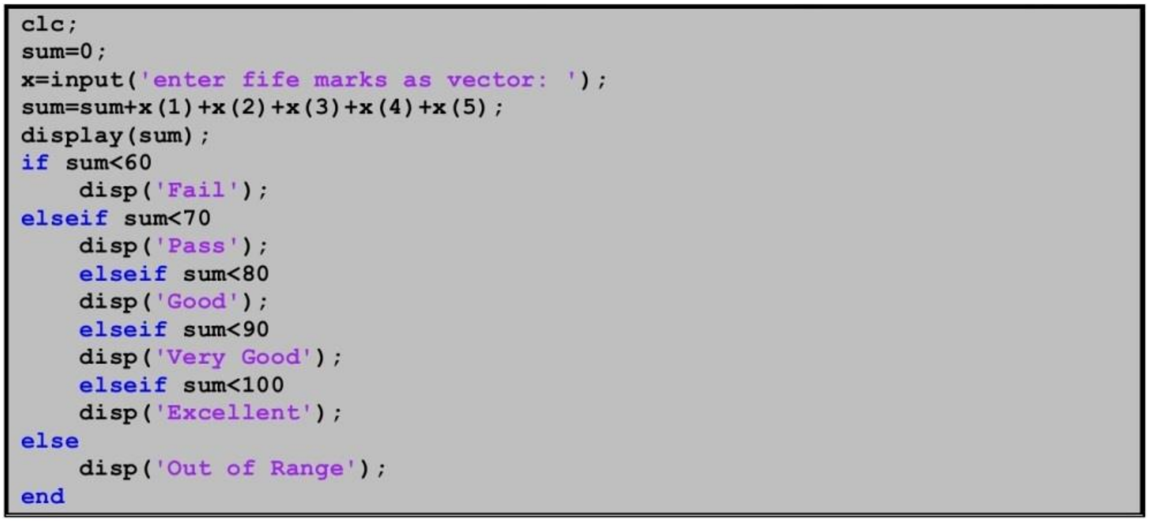
-If average less than 60% print fail

-If average less than 70% print pass

-If average less than 80% print good

-If average less than 90% print very good

-If average less than 100% print excellent



```
clc;
sum=0;
x=input('enter fife marks as vector: ');
sum=sum+x(1)+x(2)+x(3)+x(4)+x(5);
display(sum);
if sum<60
    disp('Fail');
elseif sum<70
    disp('Pass');
    elseif sum<80
    disp('Good');
    elseif sum<90
    disp('Very Good');
    elseif sum<100
    disp('Excellent');
else
    disp('Out of Range');
end
```

RELATIONAL OPERATORS

Relational operator	Meaning
<	Less than.
<=	Less than or equal to.
>	Greater than.
>=	Greater then or equal to.
==	Equal to.
~=	Not equal to.

- Relational operators compare two numbers in a comparison statement.
- If the statement is true, it is assigned a value of 1.
- If the statement is false, it is assigned a value of 0.

RELATIONAL OPERATORS, EXAMPLES

```
>> 5>8

ans =

     0

>> a=5<10

a =

     1

>> y=(6<10) + (7>8) + (5*3= =60/4)

y =

     2
```

Since 5 is not larger than 8 the answer is 0.

Checks if 5 is smaller than 10, and assigns the answer to **a**.

Since 5 is smaller than 10 the number 1 is assigned to **a**.

=1

=0

=1

LOGICAL OPERATORS

- Logical operators have numbers as operands.
- A nonzero number is true.
- A zero number is false.

Logical Operator	Name	Meaning
& Example: *A* & *B*	AND	True if both operands (*A* and *B*) are true.
| Example: *A* | *B*	OR	True if either or both operands (*A* and *B*) are true.
~ Example: ~ *A*	NOT	True if the operand (*A*) is false. False if the operand (*A*) is true.

LOGICAL OPERATORS, EXAMPLES

```
>> 3&7
```

3 AND 7.

```
ans =
    1
```

3 and 7 are both true (nonzero), so the outcome is 1.

```
>> a=5|0
```

5 OR 0 (assign to variable **a**).

```
a =
    1
```

1 is assigned to **a** since at least one number is true (nonzero).

```
>> x=-2; y=5;
```

Define variables **x** and **y**.

```
>> -5<x<-1
ans =
    0
```

Mathematically correct. The answer is false since MATLAB executes from left to right. -5<x is true (=1) and then 1<-1 is false (0).

```
>> -5<x & x<-1
ans =
    1
```

The mathematically correct statement is obtained by using the logical operator &. The inequalities are executed first. Since both are true (1), the answer is 1.

EXAMPLE OF USING THE `if-end` STATEMENT

```
% A script file that demonstrates the use of the if-end statement.
% The user is asked to enter three grades.
% The program calculates the average of the grades.
% If the average is less than 60, a massage:
% The student did not pass the course.  is printed.

score = input('Enter (as a vector) the scores of the three tests   ');
ave_grade = (score(1) + score(2) + score(3))/3;
disp('The average grade is:')
disp(ave_grade)
if ave_grade < 60
    disp('The student did not pass the course.')
end
```

EXAMPLE OF USING THE `if-end` STATEMENT

Executing the script file of the previous slide in the Command Window:

```
>> Lecture8Example1
Enter (as a vector) the scores of the three tests   [78 61 85]
The average grade is:
   74.6667

>> Lecture8Example1
Enter (as a vector) the scores of the three tests   [60 38 55]
The average grade is:
    51
The student did not pass the course.
```

EXAMPLE OF USING THE

if-else-end STATEMENT

```
% A script file that demonstrates the use of the if-else-end statement.
% The user is asked to enter three grades.  The program calculates
% the average of the grades.  If the average is less than 60, a
% massage:  The student did not pass the course.  is printed.
% Otherwise, a massage: The student passed the course. is printed.

score = input('Enter (as a vector) the scores of the three tests   ');
ave_grade = (score(1) + score(2) + score(3))/3;
disp('The average grade is:')
disp(ave_grade)
if ave_grade < 60
    disp('The student did not pass the course.')
else
    disp('The student passed the course.')
end
```

EXAMPLE OF USING THE `if–else-end` STATEMENT

Executing the script file of the previous slide in the Command Window:

```
>> Lecture8Example2
Enter (as a vector) the scores of the three tests   [65 80 83]
The average grade is:
    76
The student passed the course.
>> Lecture8Example2
Enter (as a vector) the scores of the three tests   [60 40 55]
The average grade is:
   51.6667
The student did not pass the course.
```

EXAMPLE OF USING THE
if-elseif-else-end STATEMENT

```
% A script file that demonstrates the use of the if-elseif-else-end
% statement.
% The program calculates the tip in a restaurant according to the
% amount of the bill.
% If the bill is less than 10$ the tip is $1.80.
% Between $10 and $60 the tip is 18% of the bill.
% Above $60 the tip is 20% of the bill.

format bank
clear tip
```

(The file continues on the next slide)

(Continuation from the previous slide)

```
bill = input('Enter the amount of the bill (in dollars):  ');
if  bill <= 10)
    tip = 1.8;
elseif (bill > 10) & (bill <= 60)
    tip = bill*0.18;
else
     tip = bill*0.2;
end
disp('The tip is (in dollars):')
disp(tip)
```

EXECUTING THE SCRIPT FILE OF THE RESTAURAT TIP CALCULATION

```
>> Lecture8Example3
Enter the amount of the bill (in dollars):  15
The tip is (in dollars):
          2.70
```

```
>> Lecture8Example3
Enter the amount of the bill (in dollars):  6
The tip is (in dollars):
          1.80
```

```
>> Lecture8Example3
Enter the amount of the bill (in dollars):  100
The tip is (in dollars):
         20.00
```

COMMENTS ABOUT **if–end** STATEMENTS

- For every **if** command a computer program must have an **end** command.
- A program can have many **if ….. end** statements following each other.
- A computer program can perform the same task using different combinations of **if - end**, **if – else – end**, and **if – elseif – else – end** statements.