Lesson 9 Conditional Control Statements

Exercise: 1

Use of `if` statement

```
To test whether a number is even or odd
a=input('Enter an integer: ');
if rem(a, 2) == 0 disp('Even INTEGER!');
end
```

Save the program as **`evenodd.m`** and execute in the command window as follows

```
>> evenodd

Enter an integer: 18

EVEN INTEGER!
```

Exercise: 2

The use of **`if-else-end`** STATEMENT

```
% To test whether a number is a prime or composite n = input
('Enter a natural number: ');
if isprime(n)
fprintf ('The given number %d is a prime, n);
else fprintf('The given number %d is a composite number');end
```

Save the program as `prycomp.m` and execute the same in command prompt.

```
>> prymcomp

Enter a natural number: 191 The given number 191 is a prime

>> prymcomp Enter a natural number: 637

The given number 637 is a composite number
```

Exercise 3:

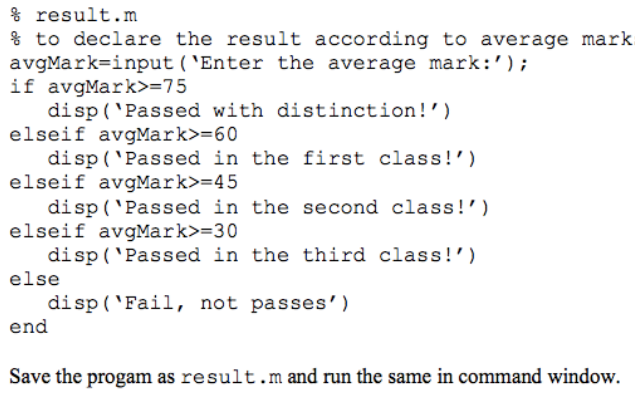
The use of **`if-elseif-elseif.........else-end`** statement

A program to declare the result of a student based on the average mark obtained by him/her.

The classification of the result is as follows:

Distinction	average mark <= 75
First Class	60 < = average mark < 75
Second Class	45 < = average mark < 60
Third Class	30 < = average mark < 45
Fail	average mark < 30

Create the following script file in the editor window.



```
% result.m
% to declare the result according to average marks obtained
avgMark=input('Enter the average mark:');
if avgMark>=75
   disp('Passed with distinction!')
elseif avgMark>=60
   disp('Passed in the first class!')
elseif avgMark>=45
   disp('Passed in the second class!')
elseif avgMark>=30
   disp('Passed in the third class!')
else
   disp('Fail, not passes')
end
```

Save the progam as `result.m` and run the same in command window.

Nested Control Statements

Loops

As the name implies, loops perform operations repetitively. There are two types of loops, depending on how the repetitions are terminated.

a. `for loop`

b. `while loop`

A `for loop` ends after a specified number of repetitions.

A `while loop` ends on the basis of logical conditions

The `for` …….`end` structure

A `for loop` repeats statements a specific number of times. Its general syntax is

```
for index = start:step:finish
      Statements
End
```

The *forloop* operates as follows. The *index* is a variable that is set at an initial value, *start*. The program then compares the index with a desired final value, *finish*. If the index is less than or equal to *finish*, the program executes the *statements*. When it reaches the *end* line that marks the end of the loop, the *index* variable is increased by the *step* and the program loops back up to the *for*statement. The process continues until the index becomes greater than the *finish* value. At this point the loop terminates as the program skips down to line immediately following the *end* statement

Exercise 4:

Use of `for loop` to compute a factorial

Develop a function file to compute the factorial

0! = 1

1! = 1

2! = 1 x 2 = 2

3! = 1 x 2 x 3 = 6

4! = 1 x 2 x 3 x 4 = 24

5! = 1 x 2 x 3 x 4 x 5 = 120

```
functionfout = factor(n)
%factor(n):input for n!; fout factorial output or result
%Computes the product of all the integers from 1 to n
x =1;
fori = 1:n
x = x*i;
end
fout=x
end
```

Save the function file by the default name factor and execute it as given below from command window

```
>>factor(6)

fout =

   720

ans =

   720
```

The `while loop` structure

A while loop repeats as long as logical condition is true. Its general syntax is

while condition

```
statements
end
```

The statement between while and end are repeated as long as the condition is true. A simple example is given below:

Exercise 5:

Develop a script file as given below:

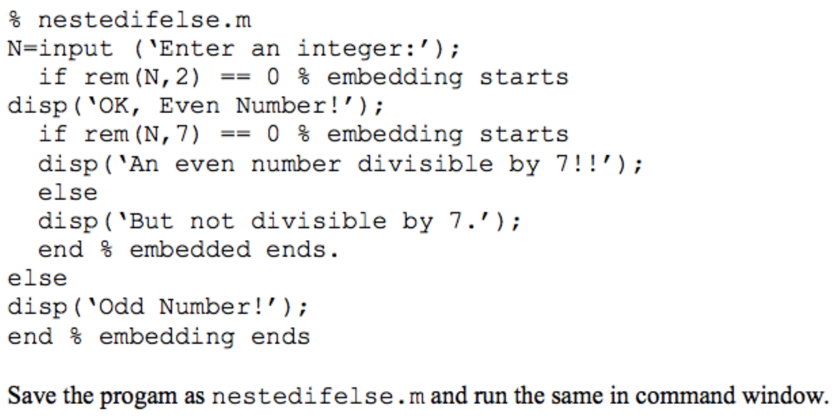
Open the editor window and type the following

```
x = 8
while  x > 0
x = x - 3;
disp(x)
end
```

Save the file by default as `untitled3` and execute the same in command window as shown below

```
>>untitled3

x =

8

     5

2

    -1
```

Exercise 6:



```
% nestedifelse.m
N=input ('Enter an integer:');
  if rem(N,2) == 0 % embedding starts
disp('OK, Even Number!');
  if rem(N,7) == 0 % embedding starts
  disp('An even number divisible by 7!!');
  else
  disp('But not divisible by 7.');
  end % embedded ends.
else
disp('Odd Number!');
end % embedding ends
```

Save the progam as `nestedifelse.m` and run the same in command window.