1-Introduction:-

1.1 simple math:-

بأستطاعة ال MATLAB أن ينفذ العمليات الحسابية البسيطة مثلما تنفذها أل calculator .

EX: - buy 4 erasers at 25 cents each
6 memo pads at 52 cents each
2 rolls of tape at 99 cents each

ما هو عدد ال items ؟ وما هي التكلفة الكلية؟

```
> > 4+6+2
    ans=12

>>4*25+6*52+2*99
  ans=610
```

في كلا الحالتين سوف يخزن النتيجة في متغير اسمه ans .
ويمكن حل السؤال أعلاه من خلال استخدام متغيرات أل MATLAB وكما يلي:-

```
>>erasers = 4 → erasers = 4
>>pads = 6   → pads = 6
>>tape = 2;
```

سوف لا يظهر شيء وذلك لوجود ألفارزه.

```
>> items = erasers+pads+tape
  items= 12

>>cost=erasers*25+pads*52+tape*99
  cost=610
```

الجدول التالي يوضح العمليات ألحسابيه الاساسيه في أل MATLAB

Operation	***Symbol***	***Example***
Addition, a+b	+	5+3
Subtraction, a-b	-	23-12
Multiplication, a.b	*	3.14*0.85
Division, a÷b	/ or \	56/8=8\56
Exponential, a^b	^	5^2

1.2 The MATLAB workspace:-

أن أل MATLAB له ألقابليه على خزن وتذكر الأوامر وكذلك قيم المتغيرات التي تم تكوينها. وأن هذه الأوامر والمتغيرات سوف تحفظ في أل MATLAB workspace ويمكن استدعاءها عند ألحاجه.

EX:-

```
>>tape
  tape=
       2
```

ملاحظه:- في حالة نسيان اسم المتغير فيمكن عرض قائمه بأسماء جميع المتغيرات ألمستخدمه بواسطة الإيعاز who

```
>>who
Your variables are
ans  cost  items  tape  erasers  pads
```

ملاحظه:-لغرض تذكر الأوامر ألسابقه يستخدم المفاتيح (↓↑→←).

1.3 About variables:-

أن أسماء المتغيرات في أل MATLAB تكون محكومه بقواعد محدده شأنها شأن باقي لغات البرمجة. بشكل عام أسماء المتغيرات يجب أن تتألف من كلمه واحده ولا تحتوي على فراغ. وقواعد تسمية المتغيرات هي:-

1. المتغيرات التي تكتب بأحرف كبيرة تختلف عن المتغيرات التي تكتب بأحرف صغيره (Items, items, iTems, ITEms) تعتبر متغيرات مختلفة.
2. اسم المتغير يحتوي على character -31 كحد أعلى ,أكثر من 31 يهمل.
3. اسم المتغير يجب أن يبدأ بحرف letter يليه عدد من الأحرف letters ,الأرقام digits ,أو أل underscore (a_b_c_d_e),(x51483)
4. لايمكن استخدام علامات التنقيط.

1.4 Comments and Punctuation:-

- كل نص يأتي بعد العلامة (%) يعتبر تعليق. وأن أل MATLAB سوف يهمل العلامة (%) وكل ما يأتي بعدها. (هذه العلامة تستخدم للتوثيق)

EX:-

```
>>erasers=4  % number of erasers
   erasers=
        4
```

- يمكن وضع عدة أوامر في سطر واحد بشرط أن تكون مفصولة بـ commas أو semicolons .

EX:-

```
>>erasers=4,pads=6;tape=2
erasers=
     4
tape=
     2
```

ونلاحظ عدم طباعة قيمة المتغير pads والذي يحمل القيمة 6 وذلك لوجود الفارزة المنقوطة.

```
>>items=erasers+pads+tape;
```

يتم حساب المجموع ويخزن في المتغير items ولكن لايطبع على الشاشة لوجود الفارزة المنقوطة ;

يتم حساب الكلفة الكلية وتخزن في المتغير cost ولكن لاتطبع على الشاشة لوجود الفارزة المنقوطة ;

```
>>cost=erasers*25+pads*52+tape*99;
```

2-Scientific Features:-

2.1 Common Mathematical Functions:-

أل MATLAB يستخدم مجموعه كبيره جداً من الدوال ألمستخدمه في التطبيقات ألرياضيه والهندسية والعلمية. ومن أهم هذه الدوال الشائعة هي:-

abs(x)	Absolute value of magnitude of complex number
acos(x)	Inverse cosine
asin(x)	Inverse sine
atan(x)	Inverse tangent
cos(x)	Cosine
exp(x)	Exponential e^x
gcd(x,y)	Greatest Common Divisor of integer x and y
lcm(x,y)	least Common Multiple of integers x and y
log(x)	Natural Logarithm
log10(x)	Common (base10) Logarithm
rem(x,y)	Remainder after Division:rem(x,y) gives the remainder of x/y
sign(x)	Signum function: return sign of argument,e.g:- sign(1.2)=1,sign(-23.4)= -1,sign(0)=0
sin(x)	Sine
tan(x)	Tangent
sqrt(x)	Square Root

مثال :- احسب ناتج ألمعادله التالية:-

$$bh=h+d.\tan(\theta)$$

```
>>h=2
   h=
      2
>>theta=6
    theta=
          6
>>d=50
  d=
     50

>>bh=h+d*tan(theta*pi/180)
bh=
   7.2552
```

Examples:-

```
>>4*atan(1)
ans=3.1416
>>180/pi*atan(-2/3)
ans=-33.69
>>y=sqrt(3^2+4^2)
y=5
>>y=rem(23,4)
y=3
>>gcd(18,81)
ans=9
>>lcm(18,81)
ans=162
```

Example:-

$$p=a\left(\frac{r(1+r)^m}{(1+r)^m - 1}\right)$$, a=18500, m=12*4, r= (2.9/100)/12

sol:-

```
>>a=18500;
>>m=12*4;
>>r=(2.9/100)/12;
>>p=a*(r*(1+r)^m/((1+r)^m-1))
    p=408.67
```

3-Command Window Features:-

3.1 Managing the MATLAB Workspace:-

أسماء المتغيرات والبيانات التي تم تكوينها في نافذة الأوامر تكون مخزونه في ما يسمى بال MATLAB Workspace. ولمعرفة أسماء المتغيرات الموجودة في أل MATLAB Workspace نستخدم الايعاز who .

```
>>who
Your variables are:-
a   m   r   p
```

ولمعرفة أسماء المتغيرات مع كافة المعلومات المتعلقة بها نستخدم الايعاز whos .

```
>>whos
Name                    Size            Bytes    Class
  a                     1*1              8       double array
  m                     1*1              8       double array
   .                     .               .          .
   .                     .               .          .
   .                     .               .          .
Grand total is      elements using    bytes
```

هذا الإيعاز سوف يقوم بعرض كل متغير مع الحجم وعدد البايتات ألمستخدمه ونوعه class ونلاحظ إن جميع المتغيرات تكون من نوع double precision ما لم يتم تحديد نوع المتغير.
ولغرض حذف المتغيرات من أل MATLAB Workspace نستخدم الإيعاز clear

```
>>clear a m          a و m لحذف المتغيرين
>>who
Your variables are:-
                              أسماء المتغيرات .....
>>clear x*          x لحذف كل المتغيرات التي تبدأ بالحرف
>> clear all
                    لحذف جميع المتغيرات والدوال functions ألمستخدمه في أل Workspace
```

3.2 Saving and Retrieving Data:-

إن أل MATLAB لـه الامكانيـه علـى خـزن وتحميـل البيانـات مـن الملفـات ألمخزونـه فـي ألحاسبه. الاختيـار save work space as من قائمة ألملفات file menu سوف يقوم بفتح ملف لخزن جميع المتغيرات. والاختيـار Import data في أل file menu سوف يقوم بفتح صندوق حوار لتحميل المتغيرات ألمخزونه.
وكذلك يمكن استخدام الأيعازين save و load لخزن وتحميل البيانات.

```
>>save
          يقوم بخزن جميع المتغيرات في MATLAB binary format في الملف matlab.mat
>>save data
                    يقوم بخزن جميع المتغيرات في الملف data.mat
>> save data1 var1 var2 var3
          يقوم بخزن المتغيرات var1,var2,var3 في الملفdata1 بصيغة أل binary format .
```

أما الإيعاز load فيقوم بتحميل المتغيرات في ال MATLAB work space

```
>>load data1
                    يقوم بتحميل جميع المتغيرات الموجودة في الملف data1
```

مثال :- نقوم بتكوين ثلاث متغيرات هي a ,b,c وتحمل قيما معينة لكل متغير وكما يلي

```
>>a=5,b=2,c=7
a =
```

```
    5


b =

    2


c =

    7
```

ثم نقوم بخزن قيم هذه المتغيرات الثلاثة في ملف اسمه fil1 وكما يلي :-

```
>> save fil1 a b c
```

ثم نقوم بحذف جميع المتغيرات وكما يلي :-

```
>>clear all
```

وللتأكد من حذف هذه المتغيرات نطبق الامر who وكما يلي :-

```
>>who
```

بعد ذلك نقوم بتحميل الملف fil1 مع متغيراته الثلاثة a,b,c وكما يلي :-

```
>> load fil1
```

سوف يتم تحميل المتغيرات الثلاثة وللتاكد من ذلك نستخدم الامر who

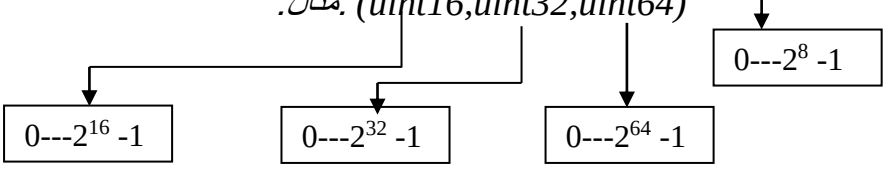
```
>>who
Your variables are:
a  b  c
```

3.3 Data Type:-

لقد تعرفنا سابقا على المتغيرات variables وما هي الشروط الواجب توفرها لتخصيص تلك المتغيرات في أل Workspace. وكذلك ذكرنا بأن تلك المتغيرات تأخذ قيما ثابتة صحيحة, عشريه أو أية أنواع بيانات أخرى. أما ألان سنستعرض أهم أنواع البيانات التي يتعامل معها MATLAB وهي:

1-العددية (صحيح integer,عشري double) هذا النوع من البيانات يأخذ الأعداد الصحيحة و ألعشريه. أن الأعداد الصحيحة يمكن أن تكون صحيحة سالبه أو موجبه لذا فهناك تعريفين أساسيين ضمن الأعداد الصحيحة هما:

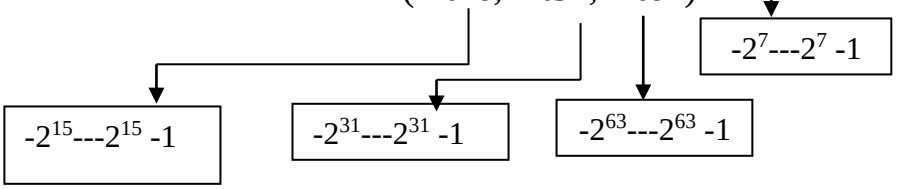
* **uint8**: هذا النوع يزود بيانات رقميه صحيحة موجبه فقط .وأن ألمساحه الخزنيه ألمخصصه لهذا النوع من البيانات هو 8-bit. اكبر رقم يمكن تخصيصه عبر هذا التعريف هو 255. وهناك تعريفات أخرى تأخذ 16,32,64-bit وهي *(uint16,uint32,uint64) .مثال:*




```
>>x=uint8(56)
x=
56
>>x1=uint8(-56)
x1=
0
>>x2=uint8(257)
x2 =
```

```
 255
>>x2=uint16(257)
x2 =
   257
```

* **int8** :هذا النوع يزود بيانات رقميه صحيحة موجبه وسالبه. أن ألمساحه الخزنيه ألمخصصه لهذا النوع من البيانات هو 8-bit. اكبر رقم يمكن تخصيصه عبر هذا التعريف هو-128..+127.وهناك تعريفات أخرى تأخذ 16,32,64-bit وهي (Int16, int32, int64) .مثال:




```
>>y=int8(-9)
y =
  -9
>>y1=int8(-200)
y1 =
 -128
>>y2=int8(180)
y1 =
  127
>>Y3=int16(180)
Y3 =
   180
```

أما بألنسبه للأعداد ألعشريه فقد زودنا MATLAB بتعريفين هما (single, double) ,حيث يستغل أل single مساحه خزنيه اقل من double ويستخدم للأعداد ألعشريه ألصغيره نسبيا.مثال:

```
>>y=-8.6
>>u=single(y)
```

2-**المنطقية (logical)** هذا التعريف يأخذ إحدى القيمتين المنطقيتين (true, false) ويستخدم عادة في عمليات ألمقارنه ألمنطقيه. يقوم MATLAB بعد تخصيص إحدى القيمتين بالاستعاضة عنهما بـ (1,0) , حيث يمثل 1 قيمة true و 0 يمثل false . مثال:

```
>>k=true
k=
1
>>k=(7>10)
k =
  0
```

3-الحرفي (char) يخصص عبر هذا التعريف قيم تأخذ واحد أو أكثر من الرموز التي تنتمي إلى:

- مجموعة الحروف الكبيرة 'A'..'Z'
- مجموعة الحروف ألصغيره 'a'..'z'
- مجموعة الرموز الخاصة ..,'-','+'

أمثله:

```
>>r='a'
r=
a
>>t='G'
r=
G
>>l='Ali'
l=
Ali
```

4-الخلايا (cells) من خلال هذا التعريف يمكن تخصيص مصفوفة ذات أبعاد مختلفة تحتوي على أنواع بيانات مختلفة (char, logical, double, integer) ضمن مساحه واحده.ويكون الوصول إلى القيم ألمخصصه لهذه الخلايا عبر ألفهرسه ألمباشره لمواقع الخلايا.

```
>>w=cell(2,2)
```
← لتكوين مصفوفة تتكون من اربع خلايا بصفين وعمودين

```
w =
    []    []
    []    []
>>w(1,1)={'a'}
```
← تخصيص القيمة a الى الخلية في الصف الاول العمود الاول من المصفوفة

```
w =
    'a'    []
    []     []
```

```
>>w(1,2)={[1 2 3]}
```
← تخصيص المصفوفة 3 2 1 الى الخلية في الصف الاول العمود الثاني من المصفوفة

```
w =
    'a'    [1x3 double]
    []          []>>w(2,1)={-3,5}
```

```
>>w(2,1)={-3.5}
```
← تخصيص القيمة 3.5- الى الخلية في الصف الثاني العمود الاول من المصفوفة

```
w =

    'a'          [1x3 double]
    [-3.5000]          []
```

```
>>w(2,2)={true}
```
← تخصيص القيمة المنطقية true (1) الى الخلية في الصف الثاني العمود الثاني من المصفوفة

```
w =

    'a'          [1x3 double]
    [-3.5000]    [       1]
```

5-القيود (struct) هذا التعريف يشابه كثيرا تعريف cell وباختلاف واحد هو يمكن الوصول إلى البيانات ألمخزنه عن طريق الأسماء وليس عن طريق ألفهرسه ألمباشره. مثال:

s=struct('Name',{Ali','M'},'Deg',{50 60})

4-Script M-files:-

بالنسبة للمشاكل البسيطة, فأن عملية تنفيذ الأوامر من خلال أل MATLAB prompt (ألطريقه ألسابقه) تكون عمليه سريعة وكفوءه. ولكن مع زيادة عدد الأوامر المطلوب تنفيذها, أو في حالة تغيير قيمة متغير أو أكثر وإعادة تنفيذ عدد من الأوامر فأن الطريقة ألسابقه تصبح غير كفوءة. ولحل هذه المشكلة فيمكن وضع مجموعة الأوامر في text file, ومن ثم يمكن للMATLAB من فتح الملف وتنفيذ مجموعة الأوامر كما لو تم إدخالها في ألMATLAB prompt. هذه الملفات تسمى script file أو m.file. ولتكوين script m-file, من قائمة الملفات file menu نختارnew ومن ثم نختار m-file.

Ex:-

لأيجاد معدل ثلاث درجات

```
% Example1.m script file for computing average of three degrees
deg1=90
deg2=50
deg3=70
avrg=(deg1+deg2+deg3)/3
```

هذا الملف يمكن خزنه كـ m-file بأسم example1.m من خلال اختيار save من قائمة الملفات. ويمكن للMATLAB من تنفيذ الأوامر الموجودة في هذا الملف عند كتابة اسم الملف في ألMATLAB prompt.

```
>>example1
   deg1=90
   deg2=50
   deg3=70
   avrg=70
```

يتم تنفيذ البرنامج عن طريق كتابة اسم البرنامج example1 في ال MATLAB prompt فيتم طبع قيم الدرجات الثلاث deg1 و deg2 و deg3 وكذلك المعدل avrg وكما في اعلاه. وعند تنفيذ البرنامج مثلا (example1.m) فأن الأوامر والمتغيرات الموجودة في أل M-file سوف تصبح جزء من ألwork space.
يوفر أل MATLAB عدد من الدوال المهمة والتي يمكن أن تستخدم في أل m-files وهي:-

disp(ans)	Display results without identifying variable names
Input	Prompt user for input
keyboard	Give control to keyboard temporarily (type *return* to quit)
pause	Pause until user presses any keyboard key
pause(n)	Pause for n seconds
waitforbottunpress	Pause until user presses mouse button or keyboard key

```
>>deg1
   deg1=90
>>disp(deg1)
   90
```

```
% Example2.m script file for computing average of three degrees
```
مثال على استخدام الدالة input
```
deg1=90;
deg2=50;
deg3=input('Enter the third degree>')
avrg=(deg1+deg2+deg3)/3
```

عند تنفيذ هذا الملف كما تعلمنا في المثال Example1.m سوف يظهر التنفيذ كما في ادناه:-
```
Enter the third degree>70
deg3=70
avrg=70
```
ملاحظه:- دالة أل input يمكن استعمالها مع العمليات الأخرى مثلا:-
```
a=input('Enter the value>')/100
Enter the value>5
a=
   0.05
```

5-File Management:-

5.1 MATLAB at startup:-

أن مسار بحث ألMATLAB (matlab path) هو عبارة عن تكوين قائمه بكل أل directories والتي يتم فيها خزن الملفات. إذا ما قمنا بتكوين directory لـm-file فأن مسار هذا أل directory يجب أن يضاف إلى matlab path, وألا فأن أل MATLAB لا يمكن له الوصول إلى الملفات المطلوبة. أن عملية فهم المسار من قبل ألMATLAB تكون حسب الشكل التالي:-

MATLAB search path

In general, when you enter>>cow, MATLAB does the following

(1) It checks to see if cow is a variable in the MATLAB workspace; if not ….
لفحص هل أن ألcow هو متغير موجود في ألworkspace

(2) It checks to see if cow is a built-in function; if not…
لفحص هل أن ألcow هو داله من نوع built-in

(3) It checks to see if an m-file name cow.m exists in the current directory, if not….
لفحص هل يوجد m-file بأسم cow.m في أل directory الحالي

(4) It checks to see if cow.m exists any where on the MATLAB search path by searching the path in the order in which it is specified.

لفحص هل أن cow.m موجود في أي مكان في مسار بحث أل MATLAB .

وفي حالة تحقق أي شرط من الشروط المطابقة أعلاه, فأن ألMATLAB سوف يقبله ويتعامل معه على أساس ألحاله المطابقة له. مثلا إذا كان cow متغير موجود في ألworkspace , فأن أل MATLAB لا

يمكن له أن يستخدم ألداله أو الأمر الذي اسمه cow. ويجب تجنب تكوين واستخدام متغيرات تحمل نفس أسماء الدوال الموجودة في ألMATLAB .

EX:-

```
>>sqrt=1.2
>>sqrt(2)
```

سوف يظهر خطاء لأن أل sqrt هو ليس داله لإيجاد الجذر ألتربيعي وإنما هو متغير يحمل ألقيمه 1.2 .
إن عمليه البحث عن المسار ألسابقه هي نفسها تكون متبعه عند استخدام الإيعاز load .
إذا كان لدينا m-files مخزونه في directory غير موجود في مسار البحث وغير موجود في أل directory الحالي, فأن ألMATLAB لايمكن إن يجد هذه الملفات. يوجد طريقتين لحل هذه ألمشكله:1- جعل أل directory المطلوب أن يكون هو ألdirectory الحالي وذلك باستخدام الأمر cd أو 2.pwd- إضافة الدليل directory المطلوب إلى مسار بحث ألMATLAB. ألطريقه ألثانيه يمكن تحقيقها بسهوله باستخدام أل path browser من الاختيار set path من قائمة الملفات File menu , أو من خلال الضغط على الزر Path Browser Button الموجود على ألcommand window toolbar. أو من خلال استخدام الأمر path أو add path
أل MATLAB يوفر عدد من الدوال والأوامر من خلالها نستطيع معرفة أسماء الملفات وحذف أل M-files ومعرفة وتغيير الدليل الحالي وغيرها من الدوال.
والجدول التالي يبين هذه الأوامر:-

File system functions

addpath dir1	Add directory *dir1* to the beginning of matlab path
cd	Show present working directory or folder
p=cd	Return present working directory in variable *p*
cd path	Change to directory or folder given given by path
delete test.m	Delete the M-file *test.m*
dir	List all files in the current directory or folder
edit test	Open *test.m* for editing ,same as open in file menu
edit path	Edit matlab path using a GUI,same as set path in file menu
exist('cow','file')	Check existence of file *cow.m* on matlab path
exist('d','dir')	Check existence of directory d on matlab path
inmem	List function M-files loaded in memory
ls	Same as dir
pwd	Same as cd
rmpath dir1	Remove directory dir1 from the matlab path
type test	Display the M-file *test.m* in the command window
what	Return alisting of all M-files and MAT-files in the current directory or folder.
which test	Display the directory path to *test.m*

6-Array Operations:-

جميع العمليات ألحسابيه ألمطبقه سابقا كانت تشمل الأرقام ألمفرده single number والتي تسمى scalars . وإذا ما رغبنا في تنفيذ نفس ألعمليه على أكثر من رقم في كل مره,فأن تنفيذ العمليات على أل scalar بشكل متكرر تعتبر عمليه صعبه ومستهلكه للوقت. ولحل هذه ألمشكله فأن أل MATLAB يحدد العمليات operations التي تطبق على data arrays .

6-1 Simple Arrays:-

لنأخذ المثال التالي:-

$x=0,0.1\pi,0.2\pi,\ldots,1.0\pi$

$y=\sin(x)$

ولحل هذه ألمشكله فأن ألMATLAB له ألقابليه على تكوين أل arrays بطريقه سهله جدا وكما يلي:-

```
x=[0 .1*pi .2*pi .3*pi .4*pi .5*pi .6*pi .7*pi .8*pi .9*pi pi]
```

```
x=
   columns 1 through 7
   0    0.3142    0.6283    0.9425                         1.8850
  columns 8 through 11
  2.1991       2.5133       2.8274       3.1416

>>y=sin(x)
y=
  columns 1 through 7
 0       0.3090      0.5878 . . . . . .. . . . . .0.9511
 columns 8 through 11
  0.8090    0.5878    0.3090    0.0000
```

لتكوين مصفوفة array في ألـ MATLAB فأننا نبدأ بالقوس الكبير 'left bracket' بعدها يتم إدخال القيم المطلوبة تفصلها الفراغات spaces أو ألفاصله commas ومن ثم إنهاء أل array بالقوس الكبير right bracket .

في المثال السابق فان ألMATLAB سوف يقوم بإيجاد أل sine لكل عنصر من عناصر المصفوفة x ووضعها في المصفوفة y .

6-2 Array Addressing :-

مثلما رأينا في المثال السابق,نلاحظ أن x يتكون من أكثر من عنصر واحد (11 قيمه تكون بشكل أعمده). وأن x هو عبارة عن array لها صف row واحد وأحد عشر عمود columns , رياضيا تسمى row vector أو one-by-eleven array .

وللوصول إلى أي عنصر من عناصر المصفوفة كما يلي:-

```
>>x(3) ----------→العنصر الثالث في المصفوفة
ans=0.6283
>>x(5) ----------→العنصر الخامس في المصفوفة
ans=0.9511
```

وللوصول إلى مجموعه من العناصر

```
>>x(1:5) ----------→العنصر الأول إلى الخامس في المصفوفة
ans=
```

0 0.3142 0.6283 ……1.2566
>>x(7:end) ----------→العنصر السابع إلى الأخير في المصفوفة
ans=
1.885 2.1991 ……3.1416
>>y(3:-1:1) ----------→ y العناصر الثالث فالثاني فألاول في المصفوفة
ans=
0.5878 0.3090 0

>>x(2:2:7)
ans=
0.3142 0.9425 1.5708

عرض العناصر الثاني فالرابع فالسادس في المصفوفة x . أي البدء بـ2 والزيادة بأثنان والتوقف عند الوصول إلى 7 .في هذه ألحاله إضافة 2 إلى 6 تساوي 8 والتي هي اكبر من 7 . لذلك فأن العنصر الثامن غير مشمول.
>>y([8 2 9 1]) ----------→
ans=
0.8090 0.3090 0.5878 0

العنصر الثامن والثاني والتاسع والاول من المصفوفة

6-3 Array Construction :-

في المصفوفة x ألسابقه قمنا بإدخال قيم المصفوفة بشكل منفرد,هذه ألطريقه قد تكون جيده إذا كانت عدد العناصر قليله,ولكن ماذا لو كان هناك المئات من القيم؟ سوف نستخدم الرمز : (colon notation) , وهناك طريقتين لأخال قيم x وهي:-

>>(0:0.1:1)*pi
x=
columns 1 through 7
0 0.3142 ……1.8850
columns 8 through 11
2.1991 ……3.1416

>>x1=linspace(0,pi,11)
x1= columns 1 through 7
0 0.3142 ……1.8850
columns 8 through 11
2.1991 ……3.1416

في ألحاله الأولى فأن ألصيغه (0:0.1:1) يكون مصفوفة تبداء بـ 0 ,زيادة بـ 0.1 وتنتهي بـ1 . وكل عنصر في المصفوفة يضرب بـ π لتكوين قيمة x . وفي ألحاله ألثانيه,فأن ألداله linspace تستخدم لتكوين قيم x1 . وأن ألصيغه ألعامه لهذه ألداله هي:-

linspace(first_value ,last_value ,number_of_values)

>>x2=logspace(0,2,11)

```
x2=
 columns 1 through 7
 1.0000   1.5849   ……..15.8489
columns 8 through 11
 25.1189   ……..100.0000
```

في هذه ألحاله, قمنا بتكوين مصفوفة x2 تبدأ بـ 10^0 وتنتهي بـ 10^2 ,وتحتوي على 11 قيمه.وان ألصيغه ألعامه لهذه ألداله هي:-

logspace(first_exponent,last_exponent,number_of_values)

```
>>a=1:5,b=1:2:9
a= 1 2 3 4 5
b=1 3 5 7 9
>> c=[b a]
c= 1 3 5 7 9 1 2 3 4 5
```

تكوين مصفوفة c تتألف من عناصر b متبوعة بعناصر a

```
>>d=[a(1:2:5) 1 0 1]
d= 1 3 5 1 0 1
```

تكوين مصفوفة d وتتألف من العناصر الأول,الثالث,الخامس من a متبوعة بثلاث عناصر اضافيه .

6-4 Array Orientation :-

جميع المصفوفات التي تم تكوينها في الامثله ألسابقه هي row vector ,وألان سوف نقوم بتكوين مصفوفة column vector وذلك بتحديد عناصر المصفوفة عنصر بعد عنصر,وتكون العناصر مفصولة بـ semicolons (;):-

```
>>c=[1;2;3;4;5]
c=
 1
 2
 3
 4
 5
>>a=1:5
a=1  2  3  4  5
```

تكوين المصفوفة a وهي row vector

```
>>b=a'
```

باستخدام ألـ transpose operator لتغير أل row vector إلى column vector .

```
>>c=b'
c=1  2  3  4  5

>>g=[1 2 3 4;5 6 7 8]
g=
  1 2 3 4
  5 6 7 8
```

تم تكوين المصفوفة g والتي لها 2 row و 4 columns (2-by-4 matrix) .في هذه ألحاله فأن أل semicolon يوجه أل MATLAB للبدء بصف جديد بين 4 و5 .

```
>>g=[ 1   2   3   4
       5   6   7   8
       9 10  11 12]
g=
  1    2    3   4
  5   6    7    8
  9  10  11  12
```

بالاضافه إلى ألـ semicolons ,فأن الضغط على Return أو Enter عند إدخال ألـ matrix فأنه يؤدي إلى البدء بصف جديد.

```
>>h=[1 2 3;4 5 6 7]
```

هنا يوجد خطاء في الايعاز حيث ان المطلوب هو تكوين مصفوفة باسم h تتكون من صفين, الصف الاول يتكون من ثلاث قيم والصف الثاني يتكون من اربع قيم وهذا لايجوز

```
??? All rows in the bracketed expression must have
 the same number of columns.
```

Scalar-Array Mathematics:-

يمكن تطبيق جميع العمليات ألرياضيه(الجمع,الطرح,الضرب والقسمة) بـ scalar معين فأنها يمكن إن تطبق بسهوله على جميع عناصر المصفوفة وكما في الامثله التاليه:-

```
>>g-2
ans=
      -1  0  1  2
       3  4  5  6
       7  8  9  10
```

طرح 2 من كل عنصر من عناصر g

```
>>2*g-1
ans=
        1    3    5    7
        9   11   13   15
       17   19   21   23
```

ضرب كل عنصر في g بـ 2 ,وطرح واحد من الناتج

Array-Array Mathematics:-

العمليات ألرياضيه (جمع,طرح,ضرب,قسمه) عندما تطبق على مصفوفات arrays من نفس الأبعاد فأنها تطبقelement-by-element مثلا:-

```
>>g
 g=
       1   2   3   4
       5   6   7   8
       9  10  11 12
>>h=[1  1  1  1;2  2  2  2;3  3  3  3]
h=
    1   1  1  1
    2   2  2  2
    3   3  3  3
    >>g+h
```

```
ans=
    2    3    4    5
    7    8    9   10
   12   13   14   15


>>ans-h
ans=
    1   2   3  4
    5   6   7   8
    9  10 11 12

>>2*g-h
 ans=
      1    3    5    7
      8   10  12   14
     15  17  19   21
```

إن الضرب والقسمة بين عناصر المصفوفتين (element-by-element) تعمل بنفس الأسلوب السابق.

```
>>g.*h
 ans=
      1    2    3    4
     10  12   14   16
     27  30   33   36
```

الرمز *. يجعل للـ MATLAB ينفذ عملية الضرب بطريقه element-by-element

```
>>g*h
???Error using ➔*
Inner matrix dimensions must agree
```

الضرب بدون وجود ألنقطه (.) يعني matrix multiplication .

```
>>g./h
ans=
    1.000    2.000   3.000   4.000
    2.500    3.000   3.500   4.000
    3.000    3.333   3.667   4.000
>>h.\g
ans=
2.0000   3.0000   4.0000
 3.0000   3.5000   4.0000
     3.0          3.3333  3.6667    4.0000
```

ألقسمه بدون ألنقطه (.) هي عملية matrix division وتختلف كليا عن العمليات ألسابقه.

```
>>g/h
warning: Rank deficient ,Rank= 1 to  1 =5.3291e-15.
ans=
      0    0    0   0.8333
```

```
    0    0    0   2.1667
    0    0    0   3.5000

>>h/g
warning: Rank deficient ,Rank= 1 to     =1.8757e-14.
Ans=
  -0.1250    0    0.1250
  -0.2500    0    0.2500
  -0.3750    0    0.3750

>>g,h
g=1    2    3    4
   5    6    7    8
   9   10   11   12

h=1    1    1    1
   2    2    2    2
   3    3    3    3

>>g.^2
ans=1     4     9     16
     25   36   49    64
     81  100  121  144
```

إيجاد مربع كل عنصر في g

```
>>g.^ -1
ans=
     1        0.5          0.33333        0.25
     0.2    0.16667    0.14286        0.125
     0.11111   0.1    0.090909      0.083333

>>2.^g
ans=
     2      4      8       16
     32     64     128     256
    512   1024  2048   4096

>>g.^h
ans=
     1       2       3     4
     25     36     49     64
    729   1000  1331  1728
```

```
>>g.^(h-1)
ans=
     1     1     1     1
     5     6     7     8
    81   100   121   144
```

6.5 Array of Ones or Zeros:-

يمكن تكوين مصفوفات تتكون من واحدات أو أصفار وكما يلي:-

```
>>ones(3)
ans=
     1   1   1   1
     1   1   1   1
     1   1   1   1
>>zeros(2,5)
ans=
     0   0   0   0   0
     0   0   0   0   0
>>size(g)
ans=
     3   4
```

ألداله size تستخدم لمعرفة أبعاد المصفوفة (عدد الصفوف والاعمده).
عند استدعاء دوال تكوين المصفوفات تحتوي على single input argument مثلا ones(n) فان ألMATLAB يقوم بتكوين مصفوفة مكونه من n-by-n من الواحدات. وعندما تستدعى باستخدام two input argument مثلا ones(r,c) فأن أل MATLAB يقوم بتكوين مصفوفة من r vectors و c columns .

6.6 Array Manipulation:-

لغرض توضيح معالجة المصفوفات في أل MATLAB نأخذ الامثله التأليه:-

```
>>a=[1 2 3;4 5 6;7 8 9]
a=
   1  2  3
   4  5  6
   7  8  9

>>a(3,3)=0
```

جعل العنصر في السطر الثالث, العمود الثالث من المصفوفة a يساوي صفر.

```
a= 1   2  3
    4   5  6
    7   8  0

>>a(2,6)=1
a=
```

```
   1   2   3   0   0   0
   4   5   6   0   0   1
   7   8   0   0   0   0
```

يجعل قيمة العنصر في الصف الثاني,العمود السادس تساوي واحد.في هذه ألحاله يتم توسيع حجم االمصفوفه حسب السؤال وملئها بقيم الاصفار.

```
>>a(:,4)=4

a=
  1  2  3  4  0  0
  4  5  6  4  0  1
  7  8  0  4  0  0
```

في هذه ألحاله يتم اعطاء ألقيمه 4 إلى جميع عناصر العمود الرابع من المصفوفة.

```
>>a=[1 2 3;4 5 6;7 8 9];
>>b=a[3:-1:1,1:3]
b=
   7   8   9
   4   5   6
   1   2   3
```

تكوين المصفوفة b من خلال عكس ترتيب الصفوف في المصفوفة a .

```
>>c=[a b(:,[1 3])]
c=
   1   2  3  7  9
   4   5  6  4  6
   7   8  9  1  3
```

تكوين المصفوفة c وذلك من خلال إضافة عناصر العمودين الأول والثالث من المصفوفة b إلى يمين المصفوفة a .

```
>>b=a(1:2,2:3)
b=
  2  3
  5  6
```

تكوين المصفوفة b من أول صفين وأخر عمودين من a

```
>>b=a(:)
  1
  4
  7
  2
  5
  8
  3
  6
  9
```

تكوين المصفوفة b وذلك من خلال اخذ أعمدة المصفوفة a الواحد بعد الأخر.

```
>>b=b.'
```

```
b=1  4  7  2  5  8  3  6  9
>>b=a
b=
   1   2   3
   4   5   6
   7   8   9
>>b(:,2)=[]
b=
        1   3
        4   6
        7   9
```

إعادة تكوين المصفوفة b وذلك من خلال إهمال العمود الثاني من المصفوفة الاصليه (حذف العمود الثاني).

```
>>b=b.'
b=1   4   7
  3   6   9
```

في هذه ألحاله,فأن الصف المرقم (i) سيكون هو العمود المرقم (i),إذن المصفوفة ذات الأبعاد 3-by-2 سوف تصبح 2-by-3

```
>>b(2,:) =[]
b=
   1   4   7
```

حذف الصف الثاني من المصفوفة

```
a=[1 2 3;4 5 6;7 8 9];
>>a(2,:)=b
a=
   1   2   3
   1   4   7
   7   8   9
```

استبدال الصف الثاني من a مع b

```
a=(2,2)=[]
??? Indexed empty matrix assignment is not allowed
```

يمكن للـ MATLAB حذف عمود أو صف كامل.ولكن لا يمكن حذف جزء من صف أو عمود.

```
>>b=a(4,:)
??? Indexed exceeds matrix dimensions.
```

لان a لا تحتوي على صف رابع.

```
>>g(1:6)=a(:,2:3)
g=
   2   4   8   3   7   9
```

تكوين row vector باسم g يتكون من جميع عناصر العمودين الثاني والثالث من a

```
a(2,:)=0
a=
```

```
   1   2   3
   0   0   0
   7   8   9
```

استبدال الصف الثاني من المصفوفة a بـ zeros .وهو مكافئ إلى الإيعاز الأتي:-

```
>>a(2,:)=[0 0 0]
a=
   1   2   3
   0   0   0
   7   8   9
>>d=[1 2 3 4;5 6 7 8;9 10 11 12]
d=
  1   2   3   4
  5   6   7   8
  9   10  11 12
>>d(2)
ans=5                                   العنصر الثاني
>>d(5)
ans=6                                   العنصر الخامس
>>d(end)
ans=12                                  العنصر الأخير
>>d(4:7)
ans=2  6  10   3                        العنصر الرابع إلى الأخير
```

بالاضافه إلى عنونة المصفوفات حسب الامثله ألسابقه, يمكن عنونة المصفوفات باستخدام أل logical array والتي تنتج من العمليات ألمنطقيه.

```
>>x=-3:3
x=
   -3  -2  -1  0  1  2  3
>>abs(x)>1
ans=
     1  1  0  0  0  1  1
```

تعطي logical array تتكون من ones عندما تكون ألقيمه ألمطلقه ل x اكبر من واحد (أي في حالة تحقق الشرط)

```
>>y=x(abs(x)>1)
y=
   -3  -2  2  3
```

في هذه ألحاله يتم تكوين المصفوفة y وذلك من خلال اخذ قيم x التي قيمها ألمطلقه تكون اكبر من واحد.

6.7 Subarray Searching:-

في أل MATLAB ألداله find تعطي الموقع subscript أو أل index عندما تكون نتيجة تنفيذ التعبير الرياضي المستخدم صحيحة true .

```
>>x=-3:3
x=
   -3  -2  -1  0  1  2  3
```

```
>>k=find(abs(x)>1)
k=
   1  2  6  7
```

تقوم بإيجاد المواقع subscripts عندما تكون abs(x)>1

```
>>y=x(k)
y=
    -3  -2  2  3
```

تقوم بتكوين المصفوفة y باستخدام المؤشرات أو المواقع الموجودة في k .(تعطي قيم عناصرالمصفوفه وليس مواقعها كما في المثال السابق).

```
>>a=[1  2  3;4  5  6;7  8  9]
a=
       1    2    3
       4    5    6
       7    8    9

>>[i,j]=find(a>5)
i=3
  3
  2
  3
j= 1
   2
   3
   3
```

في هذا المثال فأن مؤشرات الصفوف وألأعمده تخزن في i و j عندما يكون التعبير الرياضي هو true .

<u>ملاحظه:-</u>الدوال التي تعطي متغيرين في MATLAB يتم وضعها داخل أقواس مربعه على يسار أشارة المساواة. وهذا يختلف عن أسلوب معالجة المصفوفات في الامثله ألسابقه. لأن [i,j] الموجودة على يمين إشارة المساواة يقوم ببناء مصفوفة جديدة مكونه من j تضاف إلى يمين i .

6.8 Array Comparisons:-

في بعض الأحيان تكون من المهم المقارنة بين مصفوفتين مثلاً :

```
>>a=[ 1 2 3 ; 4 5 6 ; 7 8 9 ]' ,

a=
    1 4 7
    2 5 8
    3 6 9
>>b=a.*(-1).^a
  b=
      -1  4  -7
       2  -5  8
      -3  6  -9
>>c=1:9
  c=
```

```
    1  2  3  4  5  6  7  8  9
>>isequal (a,c)  ---------→  ans=0
>>isequal (a,a)  ---------→  ans=1
>>isequal (c,c ' ) ---------→ ans=0
```

الدالة isequal تعطي القيمة واحد أو True عندما تكون المصفوفات لهما تعطي نفس الابعاد وعناصرها تكون متطابقة .

```
>>ismember (a,b)
   ans=

          0    1    0
          1    0    1
          0    1    0

>>ismember(a,c)
   ans=
            1    1    1
            1    1    1
            1    1    1


>> x=0:2:20
   x=
      0 2 4 6 8 10 12 14 16 18 20

>>ismember (x,a)
   ans=
       0 1 1 1 1 0 0 0 0 0 0
>>ismember (a,x)
ans=

        0    1    0
        1    0    1
        0    1    0
```

الدالة is member تقوم بمقارنة المتغير الأول من الدالة مع الثاني لتعطي مصفوفة له نفس عدد عناصر في المتغير الأول .

```
>>intersect (a(:),b(:))      % the intersection of two arrays sorted
ans=2
     4
     6
     8

>>setdiff(a(:),b(:))         % the values in A that are not in B , sorted
   ans=
        1
```

تقاطع مصفوفتين , وتعطي نتائج مرتبة

القيم في المصفوفة a وغير الموجودة في b ,وتعطي نتائج مرتبة

```
     3
     5
     7
     9
```

```
>>setxor(a(:),b(:))        % the exclusive OR of two arrays , sorted
   ans=
     -9
     -7
     -5
     -3
     -1
      1
      3
      5
      7
      9
```

القيم في a وغير الموجودة في b والموجودة في b والغير موجودة في a , وتعطي نتائج مرتبة

```
>>union (a(:),b(:))        % all elements common to two arrays, sorted

  ans=
     -9
     -7
     -5
     -3
     -1
      1
      2
      3
      4
      5
      6
      7
      8
      9
```

جميع العناصر في كلا المصفوفتين, وتعطي نتائج مرتبة

يمكن تلخيص عمل الدوال السابقة من خلال الجدول التالي :

A○○rray Comparison	
Is equal(A,B)	*True if A and B are identical cal*
Is member(A,B)	*True where the elements of A are also elements of B*
Intersecl (A,B)	*Values in the intersection of A and B*
Setdiff (A,B)	*Values from A that are not in B*
Setxor (A,B)	*Values in the exclusive OR of a and B*

Union (A,B) *Values in the union of A and B*

6.9 Array size :-

تطرقنا سابقاً إلى الإيعاز Who والذي يقوم بعرض أسماء جميع المتغيرات التي تم تكوينها. في حالة التعامل مع المصفوفات arrays فأنه من المهم معرفة أحجام هذه المصفوفات . وفي ألـ MATLAB الإيعاز Whos يعطي معلومات أضافيه عن المتغيرات وأسماءها وحجم كل متغير وكذلك يعطي مجموع عدد البايتات وكذلك أصناف المتغيرات .

>>Whos

Name	Size	Bytes	Class
A	*3×3*	*72*	*Double array*
B	*1×3*	*24*	*Double array*
C	*4×3*	*96*	*Double array (Logical)*
ans	*1×4*	*32*	*Double array*
G	*3×4*	.	*Double array*
H	*3×4*	.	*Double array*
I	*4×1*	.	*Double array*
J	.	.	*Double array*
K	.		*Double array*
X	.	.	*Double array*
y	*1×4*	*32*	*Double array*

Grand total is 75 elements using 600 bytes

```
>>a=
      [1 2 3 4; 5 6 7 8]
  a=
     1 2 3 4
     5 6 7 8
>> s=size (A)
   s=
      2    4
```

إذا كان عدد ألـ output هو واحد فأن دالة أل Size تعطي لنا row vector وأن أول عنصر يمثل عدد الصفوف والعنصر الثاني يمثل عدد الأعمدة .

```
>>[r,c]=size(A)
 r=2         c=4
```

في حالة عدد ألـ Output هو اثنان ، فأن المتغير الأول يمثل عدد الصفوف والمتغير الثاني يمثل عدد الأعمدة .

```
>> r= size (a,1)        % number of rows
   r=2                                    تعطي عدد الصفوف
>> c=size (a,2)        % number of columns   تعطي عدد الاعمدة
    c=4
>> length (a)          ans=4
```

تعطي عدد الصفوف أو عدد الأعمدة ، أيهما أكبر

```
>> b=pi:0.01:2*pi ;
>> size (b)          ans=1        315
>> length (b)        ans=315          تعطي طول ألـ vector
 >> size ([ ])        ans=0        هذه الدالة تبين أن المصفوفة الفارغة يكون حجمها صفراً في عدد الصفوف والاعمدة
```

6.10 Multidimensional Arrays:-

في معظم التطبيقات فأننا نحتاج إلى معالجة المصفوفات ذات البعد الواحد والبعدين ولكن في بعض الحالات نحتاج إلى معالجة أي عدد من الأبعاد dimensions مثلاً

```
>> a = [ 1 0;0 1 ]
    a =1 0
       0 1
```

نقوم بتكوين ثلاث مصفوفات كل واحدة ذات بعدين (c , b , d) ثم نقوم بعملية concatenate في بعد ثالث باستخدام الدالة cat وبذلك تكون المصفوفة d لها صفان وعمودان وثلاث صفحات pages .
المصفوفة a تكون الصفحة الأولى ، المصفوفة b تكون الصفحة الثانية و c الصفحة الثالثة. الصفحة Page تمثل البعد الثالث للمصفوفة وهي أشبه بالصفحات الموجودة في الكتاب وتكون الواحدة فوق الأخرى .

```
>> b=[2 2;2 2]

    b=2 2
      2 2
>> c=[0 3;3 0]
    c=0 3
      3 0
>>d=cat (3,a,b,c)
d(:,:,1)=
    1 0
    0 1
d(:,:,2)=
    2 2
    2 2

d(:,:,3)=
    0 3
    3 0

>>size(d)
     ans=
          2 2 3
```

7-Relational and Logical Operations :-

7.1 Relational Operations:-

العمليات العلائقية في ألـ MATLAB تتضمن جميع عمليات المقارنة وهي :-

Relational Operator	Description
<	*Less than*
<=	*Less than or equal to*
>	*Greater than*
>=	*Greater than or equal to*
=	*equal to*
~=	*not equal to*

إن العمليات العلائقية Relational Operations في ألـ MATLAB تستخدم لمقارنة tow arrays من نفس الحجم أو لمقارنة array مع scalar وفي هذه الحالة فأن أل scalar يقارن مع جميع عناصر المصفوفة وفي هذه الحالة سوف يكون الناتج نفس حجم المصفوفة .

```
>> a=1:9 , b=9-a
a=
   1 2 3 4 5 6 7 8 9
b=
   8 7 6 5 4 3 2 1 0

>>tf= a>4
tf = 0 0 0 0 1 1 1 1 1
```

معرفة العناصر التي هي أكبر من4. أذا كانت a<=4 تكون النتيجة صفر ، وإذا كانت a>4 تكون واحد.

```
>>tf=(a== b)
tf=
  0 0 0 0 0 0 0 0 0
```

أيجاد العناصر في a والتي تكون مساوية إلى العناصر في b .الإشارتين = و = = لها معنيين مختلفين, فالإشارة = = تقوم بالمقارنة بين متغيرين وتعطي النتيجة واحد في حالة التساوي وصفر في حالة عدم التساوي ، أما الإشارة = فهي عملية تعيين ناتج عملية معينة إلى متغير معين .

```
>> tf = b-(a>2)
tf =

   8   7   5   4   3   2   1   0  -1
```

في هذه العملية تفحـــص a>2 وتقوم بطرح أل vector الـناتج من b. هذا المثـــال يوضـح ذلك ، أذا كان نـــاتج أل Logical operation هو numerical arrays مكون من واحدات ones وأصفار zeros فأنه يمكن استخدامه في العمليات الرياضية أيضاً .

7.2 Logical Operation

العمليات المنطقية في أل MATLAB تتضمن ما يلي :-

Logical Operator	Description
&	AND
|	OR
~	NOT

تستخدم التعابير المنطقية لدمج أو لأبطال التعابير العلائقية

Examples:-

```
>> a=1:9 ; b=9-a ;
>> tf=a>4
tf= 0 0 0 0 1 1 1 1 1
```

لمعرفة فيما اذا كانت a أكبر من 4

```
>> tf= ~ (a>4)
tf= 1 1 1 1 0 0 0 0 0
```

عكس النتيجة السابقة

```
>> tf = (a>2) & (a<6)
tf= 0 0 1 1 1 0 0 0 0
```

تعطي النتيجة ones عندما تكون a أكبر من 2 وأقل من 6 .

```
>>tf=(a>a)|(a<6)
tf =
   1    1    1    1    1    0    0    0    0
```

7.3 Relational and Logical Functions:-

بالإضافة إلى العمليات المنطقية والعلائقية السابقة ألـ MATLAB يتضمن دوال منطقية وعلائقية أخرى وكما يلي :-

- xor(x,y) Exclusive OR Operation

تعطي النتيجة ones أذا كانت x أو y هي non zero (True) وتعطي zeros أذا كانت كل من x و y هما zero (False) أو non zero (True) .

- any (x) Return one if any element in a vector x is non zero

تعطي one في كل عمود من المصفوفة x أذا كان يحتوي على عناصر non zero .

- all (x) Return one if all elements in a vector x non zero

يعطي القيمة one لكل عمود في المصفوفة x أذا كانت جميع عناصره هي non zero .

وهناك دوال كثيرة متوفرة ايضاً في الـ MATLAB والتي تقوم بعملية الفحص عن وجود قيمة أو حالة أو شرط معينة يتم البحث عنه وتعطينا نتائج منطقية ومن هذه الدوال هي :-

Test Functions

Is cell (x)	*True if argument is a cell array*
Is cell str(s)	*True if argument is a cell array of strings.*
Is char (s)	*True if argument is character string.*
Is empty (x)	*True if argument is empty.*
Is equal (A,B)	*True if A and B are identical*
Is field (S,'name')	*True if ' name ' is a field of structure S .*
Is finite (x)	*True where elements are finite .*
Is global (x)	*True if argument is a global variable.*
Is inf (x)	*True where elements are infinite.*
Is letter (s)	*True where elements are letters of the alphabet.*
Is logical	*True if argument is a logical array.*

Is member (a,b)	*True where elements of a are also in b.*
Is non(x)	*True where elements are NaNs.*
Is numeric (x)	*True if argument is a numeric array .*
Is prime (x)	*True where elements are prime.*
Is space (s)	*True where elements are white-space characters .*
Is sparse (a)	*True if argument is a sparse matrix .*
Is struct (s)	*True if argument is a structure .*

8-Control Flow:-

لغات البرمجة تقدم بعض الخصائص التي تسمح بالسيطرة على أنسيابية تنفيذ الاوامر وذلك بالاعتماد على صيغ أو هياكل أتخاذ القرارات .وأن ألـ MATLAB يتضمن أربعة صيغ أتخاذ القرار وهي :-

- For Loops
- While Loops
- If-else-end constructions
- Switch-case constructions

8.1 For Loops:-

ألـ For Loops يسمح بتكرار تنفيذ عدد من الاوامر بعدد محدد مسبقاً من المرات والصيغة العامة للـ For Loop هي :-

```
for x=array
    Commands
end
```

في هذه الصيغة ألـ commands الموجودة بين عبارتي For و end سوف تنفذ لمرة في كل عمود من المصفوفة .وفي كل تكرار iteration فأن x سوف تحدد لها قيم العمود التالي من المصفوفة أي بمعنى أخر خلال التكرار nth في حلقة التكرار x=array(:,n) مثال :-

```
>> for n=1:10
     x(n)=sin(n*pi / 10);
   end
>> x
x=
   Columns 1 through 7
   0.3090   0.5878 ……..                0.8090
   Columns 8 through 10
   0.5878   0.3090   0.0000
```

العبارة الاولى : لتكن n تساوي واحد الى عشرة ، بعدها يتم تقييم وتنفيذ جميع العبارات حتى العبارة التالية الـ end .في المرة الاولى عندما تكون n=1 ، المرة الثانية عندما تكون n=2 وهكذا حتى تصل الى n=10 ، وبعد هذه الحالة أي (n=10) فأن حلقة تكرار ألـLoop سوف تنتهي ويتم تنفيذ الاوامر التي هي بعد عبارة end والتي هي عملية عرض قيم x .

ملاحظة :- عبارة تكرار For Loop لايمكن أنهاءها عن طريق أعادة تخصيص ألـ Loop variable والذي هو (n) في المثال أعلاه في داخل حلقة ألـ For .

```
>> for n:1:10
```

```
      x(n)=sin(n*pi/10);
    n=10;
    end
>>x
x=
    Columns 1 through 7
    0.3090   0.5878  0.8090 ……..                    0.8090
    Columns 8 through 10
    0.5878   ……    0.0000
Ex:-
>> data=[ 3  9  45  6 ; 7  16  -1  5 ]
data=
    3   9  45  6
    7   16  -1  5
>> for n=data
    x=n(1) – n(2)
    end
x=                                    -4=(3-7)      من العمود الاول
    -4
x=                                    -7=(9-16)     من العمود الثاني
    -7
x=                                    46=(45-(-1))  من العمود الثالث
    46
x=                                    1=(6-5)       من العمود الرابع
    1
```

ألـ For Loops يمكن أن تكون متداخلة nested وكما يلي :-

Ex:-

نقوم بتكوين m-file زنكتب به البرنامج التالي:-

```
>> for n=1:5
        for m=5:-1:1
            a(n , m) = n ^ 2 + m ^ 2 ;
        end
      disp (n)
    end
1
2
3
4
5
>>a
a=
    2    5    10    17    26
```

```
    5   8   13   20   29
    10  13   18   25   34
    17  20   25   32   41
    26  29   34   41   50
```

ملاحظة :- يفضل تجنب أستخدام ألـ For Loops والتعويض عنها بطرق معالجة المصفوفات لحل المشكلة المعطاة مثلاً في المثال الاول ، يمكن أعادة كتابته كما يلي :-

```
>> n=1:10;
>>x=sin(n*pi/10)
x=
    Columns 1   through 7
    0.3090   0.5878  0.8090 ……..                    0.8090
    Columns 8   through 10
    0.5878    0.3090   ……    0.0000
```

نلاحظ أن الطريقتين تؤديان نفس النتيجة ، ولكن الطريقة الثانية تكون أسرع ، وتتطلب كتابه أقل Less Typing .

ولغرض زيادة السرعة أكثر ، من الافضل أن يتم تعيين المصفوفات مقدماً قبل أن يتم تنفيذ ألـ For Loop أو While Loop مثلاً في المثال:-

```
for n=1:10
     x(n)=sin(n*pi / 10);
end
```

في كل مرة يتم تنفيذ ألـ Commands الموجودة في ألـ For Loops ، وأن حجم المتغير x سوف يزداد بواحد وهذا يجعل ألـ MATLAB لأن يأخذ وقت لتخصيص مساحة خزنية أكثر لـ x في كل مرة خلال حلقة التكرار Loop ولغرض حذف أو ألغاء هذه الخطوة فأن المثال يمكن أن يعاد كتابته كما يلي :-

```
>> x=zeros(1,10);      %preallocalled  memory for x
```

تخصيص مساحة الى x مقدماً

```
>> for n=1:10
     x(n) = sin(n*pi/10);
   end
```

في هذه الحالة سوف يتم تغير قيم x(n) فقط دون الحاجة الى توسيع حجم المصفوفة .

8.2 While Loops:-

ألـ While Loop ينفذ مجموعة من ألـ Statement لعدد غير محدد من المرات والصيغة العامة لـ While Loop هي :-

```
While expression
   Commands…
End
```

ألـ Commands الموجودة بين while و end تنفذ طالما كانت قيمة ألـ expression هي True .

Ex:-

```
>> num=0 ; eps=1 ;
>> while (1+eps)>1
     eps=eps/2 ;
```

```
    num=num+1 ;
  end
>>num
num=53
>> EPS=2*eps
EPS=
    2.2204-16
```

هذا المثال يبين طريقة حساب قيمة eps ، والتي هي أصغر رقم يمكنه أن يضاف ألى واحد حتى تكون النتيجة أكبر من واحد بأستخدام دقة محددة وقد كانت حروف eps بـ upper case .في هذا المثال eps تبدأ بواحد وطالما كانت 1<(1+eps) هي True ، ألـ Commands داخل ألـ While Loop سوف يتم تنفيذها . وبما أن ألـ eps سوف تقسم على 2 بأستمرار ، فأن eps ستكون صغيرة جداً إلى حد أن أضافة eps إلى 1 سوف لن يكون أكبر من 1 .

(السبب :- هو أن الحاسبة تستخدم عدد محدد من ألـ digits لتمثيل الارقام . ألـ MATLAB يستخدم 16 digits ، لذلك ألـ eps تكون قريبة من 10^{-16}).

في هذه الحالة ستكون ناتج العملية 1<(1+EPS) هي False ويتم أنهاء ألـ While Loop . وأخيراً سوف يتم ضرب eps في 2 بسبب أخر عملية قسمة على 2 تجعل من قيمتها تكون صغيرة جداً .

8.3 if-else – end Constructions:-

في بعض الاحيان ، نحتاج الى عملية تنفيذ سلسلة من الاوامر حسب شرط معين وذلك بالاعتماد على عملية relational test . في لغات البرمجة هذا المنطق يمكن تحقيقه بوجود عدة انواع if-else-end Construction وكما يلي :-

```
if expression
   commands ……
end
```

الـ Commands الموجود بين if و end يتم تنفيذها اذا كان ألـ expression هو True .

Example:-

```
>> apples = 10 ;
>> cost = apples *25

cost =
      250
>> if apples > 5
   cost  = (1-20/100)*cost ;
   end
>> cost
cost =
       200
```

في بعض الحالات يكون هناك أكثر من بديل واحد في عبارة ألـ if-else-end Construction فتكون الصيغة كما يلي :-

```
if expression
      commands evaluated if True
else
      commands evaluated if False
end
```

فـي هـذه الصـيغة المجموعـة الاولـى مـن ألـ Commands سـوف تنفـذ أذا كـان ألـ expression هـو True .
والمجموعة الثانية من ال commands سوف تنفذ اذا كان ال expression هو False .
وعندما تكون هناك ثلاث بدائل أو أكثر فأن ألـ if-else-end construction تأخذ الشكل التالي :-

```
If expression1
    commands evaluated if expression is True
else if expression 2
    commands evaluated if expression is True
else if expression 3
    commands evaluated if expression is True
else if expression 4
    commands evaluated if expression is True
else if expression ....
    .
    .
    .
else
    commands evaluated if no other expression is True
end
```

مثال:- اكتب برنامج بلغة الMATLAB يقوم بايجاد مجموع الاعداد الزوجية المحصورة بين 1-10 .

الحل:-

نقوم بتكوين M-file وذلك عن طريق تنفيذ الايعاز m-file ← New ← ~~file~~ , ثم نقوم بكتابة البرنامج التالي:

```
close all
clear all
clc
evensum=0;
for i=2:2:10
   evensum=evensum+i;
end
evensum
```

بعد الانتهاء من كتابة البرنامج نقوم بخزن البرنامج بأسم معين وذلك عن طريق تنفيذ الايعاز file←save as سوف تظهر نافذة بأسم save as في الحقل file name نكتب الاسم الذي سوف يخزن فيه البرنامج مثلا exp1 ثم ننقر على زر save . ولغرض تنفيذ البرنامج نضغط على المفتاح f5 على لوحة المفاتيح , او من خلال تنفيذ الايعاز Run←Debug , او من خلال العودة الى شاشة برنامج ال MATLAB الرئيسية وذلك بالنقر عليها لتفعيلها ثم نكتب الاسم الذي تم خزن البرنامج به وهو exp1 في ال MATLAB prompt ونضغط على مفتاح enter فيتم تنفيذ البرنامج.

حل اخر:-

```
close all
clear all
clc
evensum=0;
for i=1:2:9
  evensum=evensum+(i+1);
end
evensum
```

مثال:- اكتب برنامج بلغة الMATLAB يقوم بايجاد مجموع الاعداد الفردية المحصورة بين 1-10.

```
close all
clear all
clc
oddsum=0;
for i=1:2:10
  oddsum=oddsum+i;
end
oddsum
```

مثال:- اكتب برنامج بلغة الMATLAB يقوم بايجاد مجموع الاعداد الفردية ومجموع الاعداد الزوجية المحصورة بين 1-10.

```
close all
clear all
clc
oddsum=0;evensum=0;
for i=1:2:10
  oddsum=oddsum+I;
 end
for i=1:2:10
     evensum=evensum+(i+1);
end
oddsum
evensum
```

حل اخر:-

```
close all
clear all
clc
oddsum=0;evensum=0;
for i=1:2:10
  oddsum=oddsum+i;
  evensum=evensum+(i+1);
end
oddsum
evensum
```

ملاحظة:- العبارة break تقوم بانهاء تنفيذ حلقات التكرار loop ومنها حلقات تكرار for .
مثال:-

```
close all
clear all
clc
oddsum=0;
for i=1:2:10
   if i>6
     break
  end
  oddsum=oddsum+i;
  end
oddsum
```

في البرنامج المجاور يمثل المتغير i كعداد للأرقام من 1 الى 10 وبزياده مقدارها اثنان .ثم يقوم البرنامج بأيجاد مجموع الاعداد (1,3,5) فقط في العبارة oddsum=oddsum+i .

سلسلة العبارات

```
if i>6
    break
end
```

وم بفحص قيمة العداد i .وعندما تكون قيمة i اكبر من 6 فيتم تنفيذ العبارة break ويتم القفز الى ما بعد حلقة التكرار .

مثلال:- اكتب برنامج بلغة ال MATLAB يقوم بعرض رسالة الى المستخدم لأدخال رقم موجب , ثم يقوم البرنامج بحساب مضروب ذلك الرقم .

الحل :-

```
close all
clear all
clc
factval=1;
n=input('Enter positive value=')
if n<0
   disp('The entered value is not posotive')
```

مثال:- اذا كانت لديك المصفوفة التالية :-

```
        -1   2   0  14  -5
         2   0   0   3  -6
a=      12 -11  10 -10   3
        -2   4   4 -16  14
         0  -2   0   2   0
```

اكتب برنامج بلغة ال MATLAB يقوم بايجاد مجموع عناصر محيط المصفوفة.

الحل:-

```
close all
clear all
clc
a=[-1 2 0 14 -5;2 0 0 3 -6;12 -11 10 -10 3;-2 4 4 -16 14;0 -2 0 2 0];
[r,c]=size(a);
s=0;
for i=1:r
    for j=1:c
        if (i==1)|(i==r)|(j==1)|(j==c)
           s=s+a(i,j);
        end
     end
end
s
```

مثال :-اذا كانت لديك المصفوفة التالية :-

```
        -6   6   5  -5   3
        -3 -12  11   4  -2
a=       2   1  12   1  -1
        -4   3  12   1  -2
        11   4   2  14  -5
```

اكتب برنامج بلغة ال MATLAB يقوم بأيجاد مجموع القيم الموجبة التي تكون قيمها خمسة فأقل , ومجموع القيم السالبة التي تكون قيمها -5 فأكبر .

الحل :-

```
close all
clear all
clc
a=[-6 6 5 -5 4;-3 -12 11 4 -2;2 1 12 1 -1;-4 3 12 1 -2;11 4 2 14 -5];
[r,c]=size(a);
ps=0;ns=0;
for i=1:r
     for j=1:c
          if (a(i,j))>0&(a(i,j))<=5
               ps=ps+a(i,j);
          elseif (a(i,j))<0 & (a(i,j))>=-5
               ns=ns+a(i,j);
          end
      end
end
ps
ns
```

مثال :- اذا كانت لديك المصفوفة التالية :-

```
    -1    3    7   17   0
     1    2   18   44   3
a= -18  -16   16    0   0
     0    1   -1    0  22
    -5    5   -6   16  33
```

اوجد مجموع القيم الموجبة وعددها ومجموع القيم السالبة وعددها .

الحل:-

```
close all
clear all
clc
a=[-1 3 7 17 0;1 2 18 44 3;-18 -16 16 0 0;0 1 -1 0 22;-5 5 -6 16 33];
sp=0;sn=0;spcount=0;sncount=0; %sp for +ve values summation,sn for _ve values
%summation,spcount for +ve values counter, sncount for –ve values counter
[r,c]=size(a);
for i=1:r
    for j=1:c
        if a(i,j) < 0
           sn=sn+a(i,j);
           sncount=sncount+1;
         elseif a(i,j)>0
           sp=sp+a(i,j);
           spcount=spcount+1;
         end
      end
end
a,sp,s spcount,n, sncount
```

حل اخر:-

```
close all
clear all
clc
a=[-1 3 7 17 0;1 2 18 44 3;-18 -16 16 0 0;0 1 -1 0 22;-5 5 -6 16 33];
sp=0;sn=0;spcount=0;sncount=0; %sp for +ve values summation,sn for _ve values
%summation,spcount for +ve values counter, sncount for –ve values counter
[r,c]=size(a);
for i=1:r
    for j=1:c
        s=sign(a(i,j));
        if s==1
           sp=sp+a(i,j);
           spcount=spcount+1;
         elseif s==-1
           sn=sn+a(i,j);
           sncount=sncount+1;
         end
      end
end
a,sp,spcount,sn,sncount
```

8.4 Switch – case Construction

صيغة ألـ Switch – Case Construction تأخذ الشكل التالي :-

```
Switch expression
        case test – expressions
              commands 1……..
        case { test – expression 2 , test – expression 3 , test – expression 4 }
              command 2 …….
        other wise
              commands 3 ……
end
```

هنا ألـ expression يجب أن تكون scalar أو character string, أذا كان ال expression هو scalar فانه يتم فحص expression= = test – expression من خلال عبارة case وأذا كان ألـ expression هو character string فأنه يتم فحص (strcmp(expression , test – expression .

في هذا المثال يتم مقارنة ألـ expression مع ألـ test – expression في أول عبارة case فأذا كانا متساويين يتم تنفيذ ألـ ……commands1 وبقية عبارات الـ case قبل عـبارة ألـ end يتم تجاوزها أو أهمـــالها . أذا كـانت أول عملية مقارنة هي not true فيـــتم الانتـقال إلـى الـثانية وفـيها يتم مـــقـــارنة ألـ expression مع test – expression 2 , test expression3 and test – expression 4

في المثال السابق فأذا كانت أي واحدة منهم تساوي الexpression ، فيتم تنفيذ Commands 2 ويتم تجاهل بقية العبارات أما أذا كانت جميع عمليات المقارنة في عبارات ألـ case هي False فيتم تنفيذ ألـ commands3 التي تنتج عبارة ألـ other wise الاختيارية .

Ex:-

```
% units.m script file
x = 2.7 ;
units = m;
      case {' inch','in'}
            y=x *2 – 54 ;
      case {'feet' , ft'}
            y= x* 2.4*12 ;
      case {'meter','m'}
            y=x/100;
      case {'millimeter','mn'}
            y=x*10;
      case {'centeter','cm'}
            y=x;
      other wise
      disp (['unknown units': units])
            d=nan;
end.
```

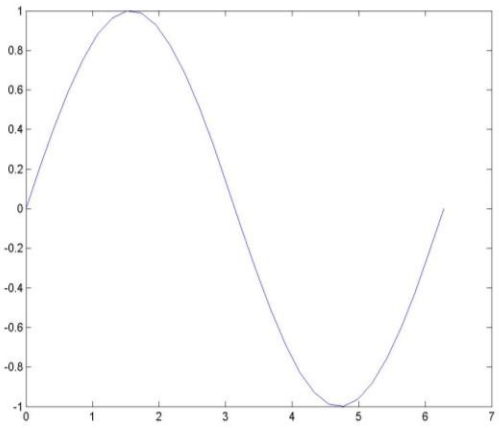
9-Two-Dimensional Graphics:-

9-1 Using the Plot Command:-

إن الإيعاز الأكثر استخداما لرسم البيانات ثنائية الأبعاد هو الإيعاز plot .هذا الإيعاز يقوم برسم مجاميع البيانات التي تكون على شكل مصفوفات array على المحاور axes ويقوم بإيصال النقاط points لتكوين خطوط معينه.(نقوم بتكوين m-file يحتوي الأيعازات التأليه):-

Ex:-

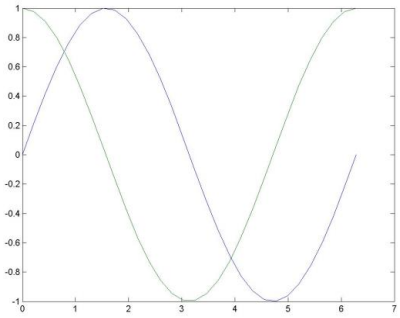
```
x=linspace(0,2*pi,30);
y=sin(x);
plot(x,y)
```



هذا المثال يقوم بتكوين 30 data points .($0 \le x \le 2$)على المحور الأفقي في الرسم, ويكون ايضاً vector أخر وهو y ويحتوي على أل sine ل data points الموجودة في x . الإيعاز plot يقوم برسم نافذة رسم تسمى بـ Graphic Window . ويقوم بعملية وضع قيم تدريجيه للمحاور بشكل مطابق مع البيانات. ومن ثم يقوم برسم النقاط وتخطيط الـ data من خلال إيصال النقاط بـ striaghtlines. ويقوم ايضاً بإضافة قيم عدديه متدرجة ووضع العلامات على المحاور.

لرسم داله الجيب ودالة جيب التمام في نفس الرسم:-

```
x=linspace(0,2*pi,30);
y=sin(x);
plot(x,y)
z=cos(x);
plot(x,y,y,z)
```

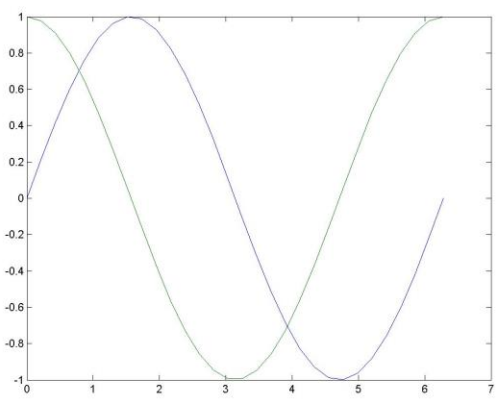


هذا المثال يوضح انه يمكن رسم أكثر من مجموعه واحده من البيانات في نفس الوقت, وذلك من خلال إعطاء أيعاز plot زوج أخر من أل argument –(x,y,x,z) .في هذا الرسم يتم رسم sin(x) مقابل أل x و cos(x) مقابل x في نفس أل plot .أيعاز أل plot يقوم برسم المنحنى الثاني بلون مختلف عن المنحنى الأول.

هذا ويمكن رسم عدة منحنيات في وقت واحد في حالة إعطاء أزواج اضافيه من أل arguments .

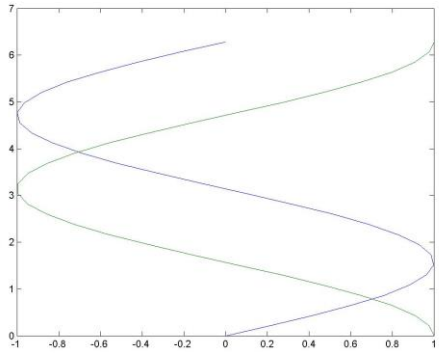
إذا كان احد أل arguments هو عبارة عن matrix والـ argument الأخر هو vector , فأن أيعاز plot سيقوم برسم كل column في الـ matrix مقابل أل vector مثلا:-

```
x=linspace(0,2*pi,30);
y=sin(x);
z=cos(x);
w=[y;z]   % create amatrix of the sine and cosine
plot(x,w) % plot the columns of w vs. x
```



إذا غيرنا ترتيب أل arguments ,الرسم سوف يدور بنسبة 90 درجه.

```
x=linspace(0,2*pi,30);
y=sin(x);
z=cos(x);
w=[y;z]   % create amatrix of the sine and cosine
plot(w,x) % plot x vs. the columns of w
```



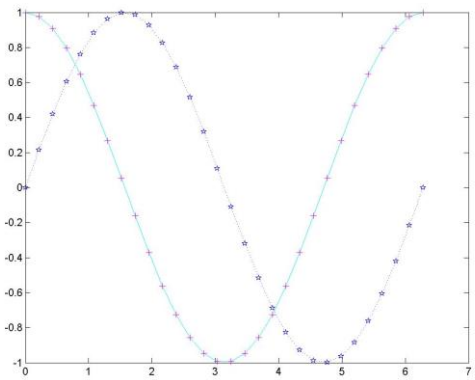
9.2 Line Style, Markers, and Colors:-

يمكن تحديد أل colors والـ markers والـ line styles وذلك من خلال إعطاء argument ثالث بعد كل زوج من البيانات. وأن هذا الاختيار الإضافي هو character string يتألف من رمز واحد أو أكثر من الجدول التالي:-

Symbol	color	symbol	marker	symbol	line style
b	blue	.	point	-	solid line
g	green	0	circle	:	dotted line
r	red	x	x-mark	-.	dash-dot line
c	cyan	+	plus	--	dashed line
m	magenta	*	star		
y	yellow	s	square		
k	black	d	diamond		
w	white	v	triangle(down)		
		^	triangle(up)		
		<	= (left)		
		>	= (right)		

إذا تم تحديد ألcolor و أل marker والـ line style ,فأن أل color يطبق على كل من أل marker والـ line .ولتحديد لون مختلف إلى أل markers ,نأخذ المثال التالي:-

```
x=linspace(0,2*pi,30);
y=sin(x);
z=cos(x);
plot(x,y,'b:p',x,z,'c-', x,z,'m+')
```



9.3 Plotting Styles:-

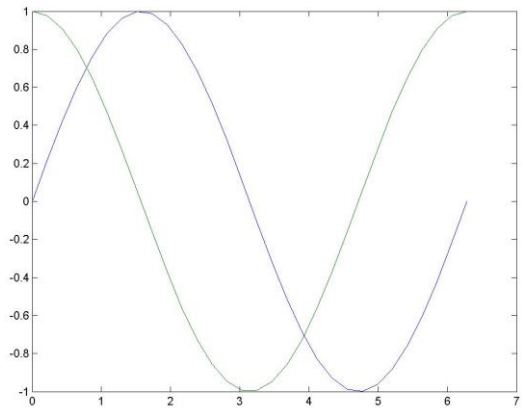
الإيعاز colordef يمكن من خلاله اختيار الشكل الكلي للرسم وكما يلي:-

-colordef white→ {*white axis background, alight gray figure background ,black axis labels,(blue ,dark green, and red)}*
-colordef black→{*black axis dackground,dark gray figure background, white axis labels,(yellow,magenta,cyan)}.*
-colordef none→*{black axis and figure background, white axis labels,(yellow, magenta, cyan)}*

9.4 Plot, Axes Box, Labels, and legends:-

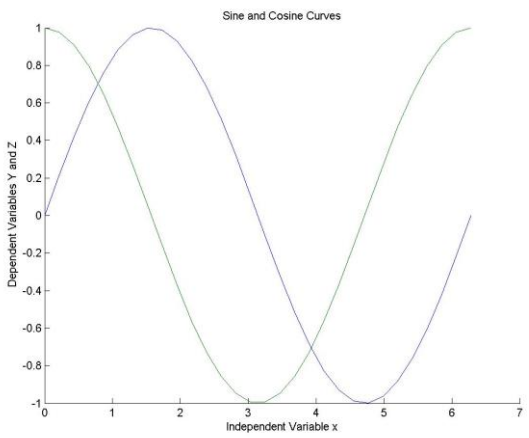
ألإيعاز grid on يقوم بإضافة grid lines إلى الرسم. Grid off يقوم بإزالة المشبك grid .الإيعاز grid لوحده يبدل من on إلى off وبالعكس. المحاور ثنائية الأبعاد two-dimensional axes توضع داخل خط مستقيم solid lines وتسمى axes box, وذلك من خلال استعمال الأمر box on , و ألأمر box off يبطل وجود أل box .المحاور الافقيه والعمودية يمكن أن تعنون باستعمال الأوامر xlabel و ylabel . والأمر title يقوم بوضع text في أعلى الرسم:-

ex:-
```
x=linspace(0,2*pi,30);
y=sin(x);
z=cos(x);
plot(x,y,x,z)
```



وألان نقوم بإزالة أل axes box, وإضافة أل title وألـ axis labels :-

```
x=linspace(0,2*pi,30);
y=sin(x);
z=cos(x);
plot(x,y,x,z)
xlabel('Independent variable x')
ylabel('Dependent variable y and z')
title('sine and cosine curves')
```



يمكن إضافة label أو text string إلى أي مكان محدد في الرسم وذلك من خلال استخدام الأمر text .وصيغة هذا الإيعاز هي:-

```
text(x,y,'string')
```

x و y يمثلان الإحداثيات coordinates من وسط جهة اليسار إلى النص.ولإضافة label لتحديد منحنى أل sine على الموقع (2.5,0.7):-

```
grid on
text(2.5,0.7,'sin(x)')
```

9.5 Printing Figures:-

يمكن طباعة الرسوم وألاشكال لـ plots وذلك باستخدام ألـ command الموجودة في شريط ألقائمه menu أو من خلال ألـ MATLAB commands .

ولطباعة الرسم plot باستخدام أل menu bar نختار الرسم وذلك بالنقر عليه بال mouse .ثم نستخدم الاختيار print من أل file menu .ثم نقوم بتحديد أل parameters الموجودة في أل print setup أو page setup فيتم إرسال الشكل إلى أل printer .

ولطباعة الشكل باستخدام أوامر الطبع في أل command window ,ننقر على الشكل بال mouse أو نستخدم الأمر figure(n) ومن ثم استخدام ألأمر print

```
print   %print the current plot to your printer
```

لتحديد اتجاه طباعة الشكل نستخدم الأمر orient وكما يلي:-

orient portrait ←*أختياري*

يطبع الشكل عموديا في وسط ألصفحه

orient landscape

يطبع الشكل أفقيا ويملئ ألصفحه

orient tall

يطبع الشكل عموديا ويملئ ألصفحه

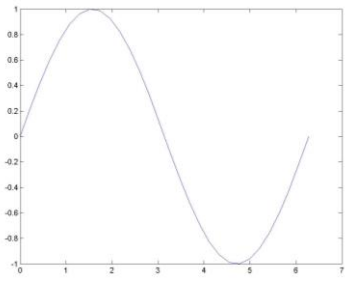
9.6 Manipulating Plots:-

يمكن إضافة lines جديدة إلى الرسم الحالي وذلك باستخدام الأمر hold. الأمر hold on فأن أل MATLAB لا يقوم بمسح المحاور الموجودة في الرسم عند استخدام أمر plot جديد ولكن سوف يقوم بإضافة منحنيات جديدة إلى المحاور الحالية.

فإذا كانت البيانات الجديدة لا تتطابق مع حدود المحاور الجديدة ,فيتم عمل rescale إلى المحاور.والأمر hold off سوف يهيئ أل current figure window إلى الرسم الجديد.الأمر hold لوحده بدون argument سيقوم بإبدال وضع hold (off-on, on-off)

ex:-

```
x=linespace(0,2*pi,30);
y=sin(x);
z=cos(x);
Plot(x,y)
```



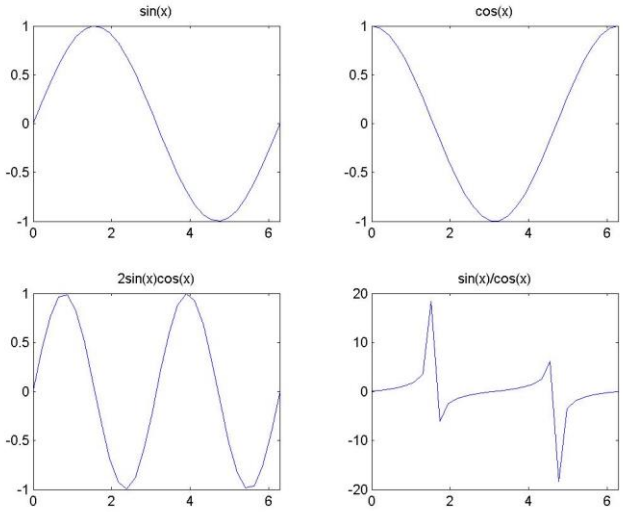
وألان نبقي الرسم ونضيف له منحنى أل cosine

```
hold on
plot(x,y,'m')
```

لمعرفة حالة أل hold إذا كانت on أو off نستخدم ألداله is hold ,فإذا كانت النتيجة 1 يعني أن أل hold هي on وإذا كانت 0 يعني hold هي off .

إذا أردنا رسم رسمان أو أكثر من أل plots في figure windows مختلفة ,الأمر subplot(m,n,p) سوف يقوم بتجزئة أل figure window الحالي إلى مجموعه من مساحات الرسم بعدد m-by-n ويتم وضع الرسم المحدد في ألمنطقه p .مثال:-(نقوم بكتابة أل M-file التالي)

```
x=linspace(0,2*pi,30);
y=sin(x);
z=cos(x);
a=2*sin(x).*cos(x);
b=sin(x)./(cos(x)+Eps);
subplot(2,2,1)
plot(x,y),axis([0 2*pi -1 1]),title('sin(x)')
subplot(2,2,2)
plot(x,z),axis([0 2*pi -1 1]),title('cos(x)')
Subplot(2,2,3)
plot(x,a),axis([0 2*pi -1 1]),title('2sin(x)cos(x)')
subplot(2,2,4)
plot(x,b),axis([0 2*pi -20 30]),title('sin(x)/cos(x)')
```



10-Text:-

أن قوة أل MATLAB، هي قابليته على معالجة الأرقام .وفي بعض الأوقات يكون من المهم معالجة النصوص Text. المتغيرات النصية في أل MATLAB تسمى بال Character String أو بال String فقط .

10.1 Character Strings :-

Example:-

```
>t='How about this character string ?'
>t=
How about this character string ?
>size(t)
ans=
     1      32
```

```
>whos
Name        size        Bytes        Class
 t           1*32        64           char
Grand total is 32 elements using 64 bytes
```

السلسلة الحرفية هي عبارة عن نص يحصر بين علامتي اقتباس مفردة. وكل حرف في السلسلة هو عنصر واحد في المصفوفة, وكل عنصر يمثل 2-Bytes .

في الأمثلة التالية سوف نقوم بتنفيذ بعض العمليات الرياضية على الـ String

```
   >double (t)
ans=
  columns 1 through 12
 72  111  119   32   97   98   111   117….
Columns 13 through 24
 105   115   …
Columns 25 through 32
 32    115 ……63
   >abs(t)

ans=
 columns 1 through 12
 72 . . . . . 104
 Columns 13 through 24
 105 ……114
 Columns 25 through 32
 32 . . . . . .63
```

والدالة Char تعطي عكس التحويل

```
 >char (t)
 ans=
 How about this character string?
```

وبما إن أل string في حقيقتها هي مصفوفات عددية مع بعض الخصائص الخاصة , فأنه يمكن معالجتها باستخدام أدوات معالجة المصفوفات المتوفرة في الـ MATLAB مثال :-

```
 >u=t(16:24)

 u=
 character
 >u=t(24:-1:16)
 u=retcarahc

 >u=t(16:24)'
 u=
 c
 h
 a
```

r
a
c
t
e
r

باستخدام أل transpose operator يتم تحويل كلمة character إلى column .
إن عملية دمج السلسلة الحرفية هو مثل عملية دمج أل array , مثال :-

```
>u='my name is Ahmed.'
>v='Ahmed is my name';
>w=[u v]
w=
my name is Ahmed. Ahmed is my name
>disp(u)
  my name is Ahmed.
```

السلسلة الحرفية يمكن أن تأخذ عدة سطور rows , ولكن كل سطر يجب أن يكون له نفس العدد من الأعمدة columns .وفي هذه الحالة فأننا سوف نحتاج إلى الفراغات لجعل جميع السطور بنفس الطول .

```
>v=['character string having more than    '
     'one row must have the same number'
     'of columns just like matrices!          ']
v=
character string having more than
one row must have the same number
of columns just like matrices !

>w=['This';'does not';'work!']
??? all rows in the bracketed expression must have the same number of columns.
>w=char('However,this','does work!')
w=
however, this
does work!
>size(w)
ans=2    13
```

إن الدالة char تقوم بتكوين string array من سلاسل معينة , وتقوم بعملية توسيع السلاسل بواسطة استخدام الفراغات blanks لكي تكون المصفوفات صحيحة أي متساوية الأبعاد .

10.2 String Conversions :-

أل MATLAB يقدم مجموعة من دوال تحويل أل string ومن أمثلتها :-

Function	Description
bin2dec	Binary string to decimal integer
char	String or ASCII (double) to character array (char)
dec2bin	Decimal number to binary string
dec2hex	Decimal number to hexadecimal string
double	String or character array (char) to ASCII (double)

hex2dec	Hexadecimal string to decimal integer
hex2num	Hexadecimal string to IEEE floating point number
int2str	Integer to string
num2str	Number to string
str2num	String to number

نقوم بكتابة البرنامج التالي

```
% conversion string
rad=2.5
area=pi*rad^2
t=['A circle of radius 'num2str(rad)…
    'has an area of 'num2str(area)'.'];
disp(t)
```

بعد تنفيذ هذا البرنامج سيكون ناتج التنفيذ

A circle of radius 2.5 has an area of 19.63.

10.3 String Functions :-

أل MATLAB يوفر عدد من دوال معالجة السلسلة الحرفية ومن هذه الدوال :-

Function	Description
blanks(n)	Return a string of n blanks or spaces
deblank(s)	Remove trailing blanks from a list
findstr(s1,s2)	Find one string with in another
ischar(s)	True if input is a string (char)
isletter(s)	True where alphabet character exist
isspace(s)	True where white-space character exist
lower(s)	String to lower case
strcat(s1,s2,…)	Horizontal string concatenation
strcmp(s1,s2)	True if strings are identical
strrep(s1,s2)	Replace one string with another
strvcat(s1,s2,…)	Vertical string concatenation
upper(s)	String to uppercase

Ex:-

```
b='Ahmed and Hashim are friends';
>findstr(b,' ')
ans=
  5   10 17 21
  >findstr(b,'d')
  ans=5 9  27
  >find(b= ='p')
  ans=
     5 9 27
```

```
>findstr(b,'Ali')
ans=
[]
```

10.4 Cell Arrays of String

لقد رأينا سابقا انه جميع الصفوف في أل Arrays يجب أن يكون لها نفس العدد من الأعمدة, وهذا يسبب مشاكل في بعض الأوقات. وهذه المشاكل يمكن حلها بواسطة استخدام أل cell arrays . وفي هذا النوع يتم وضع كل نوع من أنواع البيانات في خلية Cell, وتستخدم بشكل كثير مع السلاسل الحرفية. أل cell array هي عبارة عن نوع من البيانات التي تسمح لتسمية ومعالجة مجموعة من البيانات بأحجام وأنواع مختلفة وببساطة.

ex:-

```
>c={'How';'about';'this for a';' cell array of strings ?'}
c=
'How'
'about'
'this for a'
'cell array of strings?'
>size(c)
ans=
     4     1
```

نستخدم الأقواس المتوسطة لتكوين أل cell arrays .في هذا المثال, نلاحظ أن كل عنصر في أل cell array يحتوي على سلسلة حرفية وبأطوال مختلفة.

```
>c(2:3)
ans=
  'about'
  'this for a'
>c([4 3 2 1])
ans=
      'cell array of strings?'
      'this for a'
      'about'
      'How'
>c(1)
ans=
     'How'
```

لاستدعاء محتويات خلية معينة:-

```
>s=c{4}
s=
   cell array of strings?
>size(s)
```

```
    ans=
          1     22
```

لاستخلاص أكثر من خلية واحدة , نستخدم الدالة deal :-

```
>[a,b,c,d]=deal(c{:})
a=
How
b=
about
c=
 this for a
d=cell array of strings?
>c{:}
```

تعني كل الخلايا , وهي نفس

```
>[a,b,c,d]=deal(c{1},c{2},c{3},c{4})
a=How
b=about
c=this for a
d=cell array of strings?
```

الدالة char تحول محتويات أل cell array إلى string array

```
>s=char(c)
s=How
  about
  this for a
cell array of strings?
>size(s)
 ans=4    22
>>ss=char(c(1:22))
>ss=
     How
     about
>size(ss)
ans=2     5
```

والعملية المعاكسة لها تنفذ باستخدام الدالة cellstr

```
>cellstr(s)
ans=
      'How'
      'about'
      'this for a'
      'cell array of strings?'
```

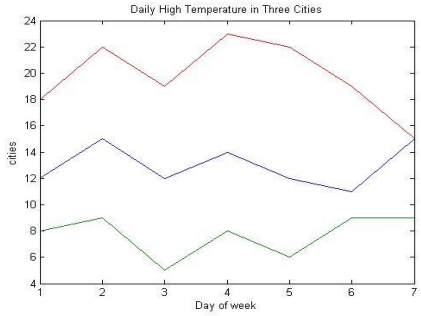
11-Data Analysis :-

يمكن للـ MATLAB أن يقوم بتنفيذ التحليلات الإحصائية المختلفة على مجاميع البيانات . مثال لنأخذ درجات الحرارة اليومية المسجلة لثلاث مدن ولمدة سبعة أيام ويتم خزنها في متغير وليكن اسمه temps ونقوم بكتابة M-file بالخطوات التالية:-

ملاحظة:- المتغير temps يخزن درجات الحرارة على شكل صفوف وأعمدة , كل صف يتضمن درجات الحرارة ليوم محدد , وكل عمود يتضمن درجات الحرارة لمدينة معينة , ثم نقوم برسم البيانات

```
temps=[12 8 18;15 9 22;12 5 19;14 8 23;12 6 22;11 9 19;15 9 15]
d=1:7
plot (d,temps)
xlabel('Day of week'),ylabel('cities')
title('Daily High Temperature in Three Cities')
```

بعد تنفيذ البرنامج السابق , سوف يتم رسم الشكل أدناه .



المتغير d هو عبارة عن متجه بطول 7, والـ temps هي مصفوفة 3-by-7 . من هذه البيانات , الإيعاز plot سوف يقوم برسم كل من المتغير temps إلى d .
ولتوضيح بعض إمكانيات ألـ MATLAB, سوف نأخذ الأمثلة التالية:-

```
>avg_temp=mean(temps)
avg_temp=
           -----------      -----------      ----------
```

في هذا المثال , فأن أل mean سوف تقوم بإيجاد معدل كل عمود على حدة . ويستطيع حساب معدل الدرجات الثلاثة أيضا:-

```
avg_avg=mean(avg_temp)
```

لحساب المعدل الكلي لدرجات الحرارة للمدن الثلاث .
وكذلك يمكن تحديد عمود معين للعمل عليه فقط , مثلا :-

```
>avg_temp=mean(temps, 1)
avg_temp=
          ------   ------   ------
```

لحساب المعدل لكل عمود

```
>avg_tempr=mean(temps,2)
avg_tempr=
           ------
           ------
           ------
          ------
          ------
```


تقوم بحساب المعدل لكل لكل سطر .

ولو أردنا حساب معدل الانحراف اليومي عن معدل درجة الحرارة لكل مدينة. فأن avg_temp(i) يتم طرحه من العمود i لكل temps . فإذا كتبنا الإيعاز:-

```
temps-avg_temp
```

سوف يظهر لنا خطاء

```
??? Error using= =>-
matrix dimensions must agree
```

وذلك لان المصفوفة temps هي 7-by-3 والمصفوفة avg_temp هي 3-by-? . ونستخدم أل for loop :-

```
>for i=1:3
        tdev(:,i)=temps(:,i)-avg_temps(i);
end
>tdev
        ------  ------  ------
        ------  ------  ------
        ------  ------  ------
        ------  ------  ------
        ------  ------  ------
        ------  ------  ------
        ------  ------  ------
>max_temp=max (temps)
  max_temp=
                ------  ------  ------
```

تقوم بإيجاد أعلى درجة حرارة لكل مدينة خلال الأسبوع .

```
>[max_temp,x]=max(temps)
max_temp=
              ------  ------  ------ ←———— أعلى درجات الحرارة

x=------  ------  ------ ←———— اليوم
```

في هذه الحالة max_temp يمثل درجات الحرارة القصوى , x يمثل اليوم الذي يسجل فيه درجات الحرارة .

```
>min_temps=min (temps)
  min_temps=
                ------  ------  ------
```

تقوم بإيجاد درجات الحرارة الصغرى لكل مدينة

```
>[min_temp,n]=min(temps)
   min_temp=
                ------  ------  ------
   n=
           ------  ------  ------
```

أل min_temps يمثل درجات الحرارة الصغرى , n يمثل اليوم الذي تسجل فيه درجات الحرارة .

```
>s_dev=std(temps)
```

s_dev=

------ ------ ------

تقوم الدالة std بإيجاد أل standard deviation .

>dialy_change=diff (temps)
dialy_change=

------ ------ ------
------ ------ ------
------ ------ ------
------ ------ ------
------ ------ ------
------ ------ ------
------ ------ ------

هذا الإيعاز يقوم بحساب التغيير في درجات الحرارة من يوم ليوم . مثلا أول صف هو التغيير في درجات الحرارة بين اليوم الأول والثاني.

11.1 Data Analysis Functions :-

يحتوي أل MATLAB على مجموعه من الدوال وتتضمن :-

Function	Description
diff(x)	Compute differences between elements
max(x),max(x,y)	Maximum component
max(x,n)	Maximum along dimension n
mean(x)	Mean or average value of columns
mean(x,n)	Mean along dimension n
median(x)	Median value of columns
median(x,n)	Median along dimension n
min(x),min(x,y)	Minimum component
min(x,n)	Minimum along dimension n
prod(x)	Product of elements in columns
prod(x,n)	Product along dimension n
rand(x)	Uniformly distributed random numbers
randn(x)	Normally distributed random numbers
sort(x)	Sort columns in ascending order
sort(x,n)	Sort along dimension n
sort rows(A)	Sort rows in ascending order
std(x)	Standard deviation of columns
sum(x)	Sum of elements in each column
sum(x,n)	Sum along dimension(n)