لغة البرمجة MATLAB : The MATLAB Programming Language

المقدمة :

هو البرنامج الهندسي الأكثر استخداماً حول العالم فيما يتعلق بالحسابات الرياضية، الهندسية والمحاكاة. والبرنامج العملاق هذا، يستخدم في الصناعات المختلفة كما يستخدم للأغراض الأكاديمية، وخصوصاً أغراض البحث العلمي في الغالبية العظمى في جامعات العالم أتت تسمية هذه اللغة من اختصار التعبير MATrix LABoratory (مختبر المصفوفة)، حيث إن البرنامج مصمم أساسا للتعامل مع العمليات على المصفوفات بشكل بسيط. كما أرفقت بهذه اللغة أدوات لمعالجة وحل تطبيقات علمية خاصة سميت toolboxes (وهي أكثر من عشرين أداة)، وتعتبر هذه الأدوات هامة جداً لمستخدمي هذه اللغة، حيث تسمح لهم بتعلم وتطبيق تقنيات حل متخصصة لمعالجة مشكلات ومسائل خاصة، مثل معالجة الإشارة، ونظم التحكم والمحاكاة والشبكات العصبية والتحليل الكمي والمالي والإحصاء ومسائل الجبر الخطي ... الخ. يؤمن برنامج MATLAB أدوات واجهة التخاطب الرسومية Graphical User Interface (GUI) التي تجعلك تتعامل مع البرنامج على انه أداة تطبيقية متطورة.

استخدامات برنامج MATLAB :

- إنجاز الحسابات (Math and Computation)
- تطوير الخوارزميات (Algorithm Development)
- تحصيل المعطيات (Data Acquisition) وهو ما يسمى بـ { الاستيفاء الرياضي}
- النمذجة و المحاكاة و إعادة نمذجة النظام (Modeling , Simulink , and Prototyping)
- تحليل المعطيات و معالجتها و إظهار النتائج على شكل رسومات (Data Analysis Exploration , and Visualization)
- محاكاة للعلوم الهندسية و الفيزيائية و ذلك عن طريق الرسومات.
- تطوير التطبيقات (Application Development)

إن MATLAB هو نظام تفاعلي ، يتعامل مع المعطيات على أنها مصفوفات و لا داعي لتعريفها أو تحديد أبعادها ، و هذه الطريقة تسمح لنا بحل الكثير من المسائل و المشكلات التقنية و خصوصا تلك التي لها شكل مصفوفات وا سعة وأنساق ، كما يمكننا أن نكتب برنامج بلغة (غير تفاعلية) مثل لغة الـ (C) أو لغة الـ (Fortran) و في الإصدارات الجديدة من MATLAB أصبح بإمكاننا كتابة واستخدام مكتبات لغة الـ C) (و الـ (C++)

كما يتميز MATLAB بوجود برامج و أدوات خاصة لحل مشكلات و مسائل تقنية خاصة ، هذه البرامج و الأدوات الخاصة موجودة في مجموعة تدعى (Toolbox) ، إن الـ (Toolbox) هو عبارة عن توابع ودوال في MATLAB ملفات (M_Files تستخدم بيئة MATLAB لحل مشاكل و مسائل تقنية خاصة)

نظام MATLAB :

يتكون نظام MATLAB من عدة أقسام رئيسية:

1- Development Environment :

و هي عبارة عن مجموعة من العناصر و الأدوات التي تساعدنا في استخدام ملفات و توابع ودوال MATLAB ، و معظم هذه العناصر و الأدوات هي من النوع (GUI) أي (Graphical User Interface) وهي عبارة عن واجهات تخاطب مع المستخدم (واجهات تحتوي على أزرار و قوائم منسدلة وغيرها من العناصر الموجودة في لغات البرمجة المرئية) أما العناصر التي يتكون منها Development Environment فهي :

- **MATLAB Desktop**
- **Command Window**
- **Command History**
- **Editor**
- **Debugger**
- **Help Browser**
- **Workspace Browser**
- **Files Browser**
- **The Search Path Browser**

2- The MATLAB Mathematical Function Library :

هي عبارة عن مجموعة كبيرة من التوابع و الدوال الرياضية و الخوارزميات ، تبدأ من الدوال و التوابع القادرة على إنجاز الحسابات الأساسية السهلة (كالضرب و الطرح و الجمع) و تكتمل حتى نصل إلى الدوال و التوابع القادرة على إنجاز حسابات معقدة مثل (إيجاد مقلوب مصفوفة من أي درجة كانت و إيجاد تابع بيسل و إنجاز تحويل فوريه السريع)

3- Graphics :

يحتوي MATLAB على طيف واسع من التوابع و الدوال التي تمكننا من إظهار المعطيات المخزنة في مصفوفة على شكل رسم ، تقسم هذه الدوال و التوابع إلى نوعين:

- **دوال و توابع عالية المستوى :** هذه التوابع و الدوال قادرة على الرسم في المستوى ثنائي البعد (X0Y) و ثلاثي الأبعاد (XYZ) ، و قادرة أيضا على معالجة الصورة و غيرها من العمليات الرسومية المعقدة.

- **دوال و توابع منخفضة المستوى :** هذه التوابع و الدوال مخصصة للتحكم الكامل بخصائص الرسومات و بناء واجهة التخاطب مع المستخدم (GUI) في تطبيقاتنا الخاصة.

4- The MATALB Application Program Interface (API) :

هذه المكتبة تسمح لنا بكتابة برامج بلغة الـ (C) أو الـ (Fortran) في MATLAB ، تحتوي هذه المكتبة على تعليمات تستدعي برامج فرعية في MATLAB ، و تعليمات تمكننا من جعل MATLAB كنواة حسابية في برامجنا المكتوبة باللغات السابقة ، و تعليمات تمكننا من قراءة و كتابة ملفات من النوع (MAT_Files) و هي عبارة عن ملفات مخصصة للربط بين برنامج الـ MATLAB واللغات الأخرى.

MATLAB Desktop :

يظهر MATLAB Desktop عندما نشغل برنامج MATLAB ، وهو مبني بواسطة الأداة GUI و يمكننا MATLAB Desktop من كتابة البرامج بلغة MATLAB وبناء تطبيقاتنا الخاصة ، و ذلك من خلال (Command Window) ، كما يمكننا من رؤية حالة و خصائص المتغيرات المستخدمة في برنامجنا من خلال (Workspace) ، كما يمكننا من رؤية التعليمات التي استخدمناها مؤخرا في برنامجنا ووقت وتاريخ بدء كتابة البرنامج و ذلك من خلال Command History كما يمكننا من رؤية الملفات المستخدمة في برنامجنا و تطبيقاتنا و ذلك من خلال (Current Directory) ، سيأتي الحديث لاحقا عن الـ MATLAB Desktop بالتفصيل .

الصناعات التي يغطيها ماتلاب :

أولاً: الفضاء الجوي والدّفاع Aerospace and Defense

الكثير من شركات الطيران المدني والعسكري، تستخدم ماتلاب في الحسابات الهندسية، النمذجة والمحاكاة. ومثال على ذلك شركة إيرباص. كما ويُعتمد على ماتلاب في تصميم الطائرات التي تطير بدون طيّار، أضف إلى ذلك، استخدام ماتلاب في أبحاث الفضاء من شركة ناسا. ويعتبر

هذا الاستخدام لماتلاب واحد من أكثر الاستخدامات تطوراً وتعقيداً.

ثانياً: صناعة السياراتAutomotive

صناعة السيارات هي واحدة من أهم وأعقد الصناعات في العالم، ولتوفير الجهد والوقت، تستخدم الشركات المصنعة برنامجي (MATLAB و Simulink الذي سنتعرف عليه لاحقاً) لخلق أو تطوير النماذج الهندسية قبل تطبيقها على أرض الواقع. فبعد اختبار هذه النماذج بواسطة ماتلاب، يتم انتاجها صناعياً. إن استخدام حزمة ماتلاب يقلل الوقت على المصممين لهذه السيارات بنسبة تصل إلى 50.%

ثالثاً: المعالجة بالتكنلوجيا الحيوية، الأدوية، والصناعات الطبية ,Biotech

Pharmaceutical, and Medical

هُنا، يستخدم ماتلاب، في معالجة البيانات من قبل الباحثين والمختصين في هذا المجال. يتم إدخال البيانات المتعلقة بالتجارب المختبرية على ماتلاب، ومن ثم تتم معالجتها بسرعة قياسية،بهذا، يتمكن الباحث من إجراء مئات التجارب الأمر الذي يستحيل فعله بطريقة يدوية.

رابعاً: الاتصالاتCommunications

علم الاتصالات، هو واحد من أحدث علوم العصر، ويمثل الثورة المعلوماتية الكبرى التي يشهدها العالم. ماتلاب يغطي علم الاتصالات وكل العلوم المتعلقه به: معالجة الإشارة الرقمية Digital Signal Processing DSP, هندسة الصوت Audio Engineering, هندسة بيانات الاتصال Data Communication Engineering, هندسة الشبكات Network Engineering.

وبشكل عام، يستخدم ماتلاب في هذا المجال لدعم نقل الصوت، البيانات، والفيديو بأقل حجم وتكلفه وبأفضل جودة. كذلك في تحليل ومحاكاة شبكات الهاتف المحمول GSM/TDMA/CDMA وغيرها، وكذلك الأمر بالنسبة لشبكات الكمبيوتر السلكية واللاسلكية. وحيث أن تقنيات النقل اللاسلكي في شبكات الكمبيوتر WiFi/WiMax هي واحدة من أكثر مواضيع الاتصالات بحثاً في يومنا هذا، فإن ماتلاب حاضر فيها وبقوة. الأمر كذلك بالنسبة لمعالجة الإشارة الرقمية، وكذلك الصورة الرقمية. إن استخدام حزمة ماتلاب في هذه الصناعات تقلل كثيراً من مخاطر التصميم.The Design Risk

صناعات أخرى يغطيها ماتلاب:

الحواسيب الشخصية (العتاد المادي)

الإلكترونيات

الخدمات والحسابات التجارية والإحصائية

الميكانيك الهندسي

صناعة أشباه الموصلات Simiconductors

القياسات الهندسية

إلكترونيات القوى

الطاقة الكهربائية

الهندسة الكهربائية

أنظمة القوى الكهربائية

أنظمة التحكم

وبالطبع، علوم الرياضيات المختلفة

وغيرها الكثير

تشغيل برنامج MATLAB

يتم تشغيل البرنامج بأحد الطرق التالية:

1- بعد تنصيب برنامج MATLAB على الحاسبة التي تعمل عليها. يتم إضافة رمز أيقونة البرنامج على سطح مكتب الحاسبة ويحمل الرمز

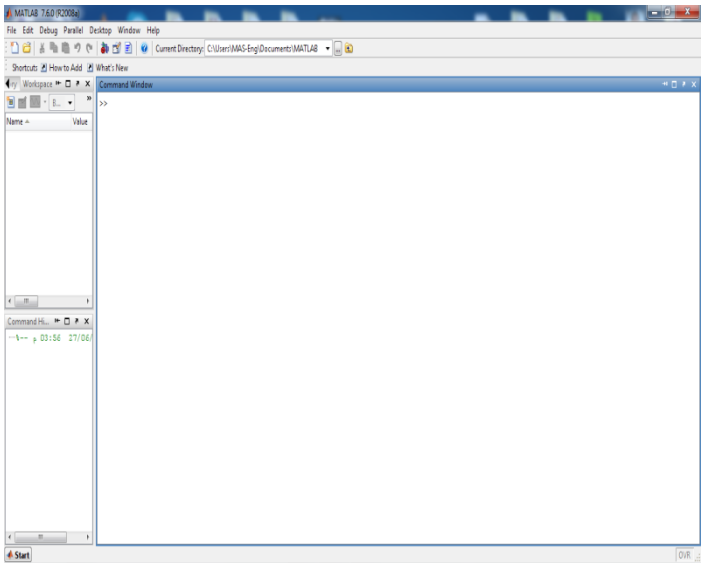
ويتم فتحة عند النقر على الأيقونة بنقرتين مزدوجتين double click.

2- أو عن طريق الذهاب إلى قائمة start ومنها إلى برامج Programs ثم أسم البرنامج MATLAB

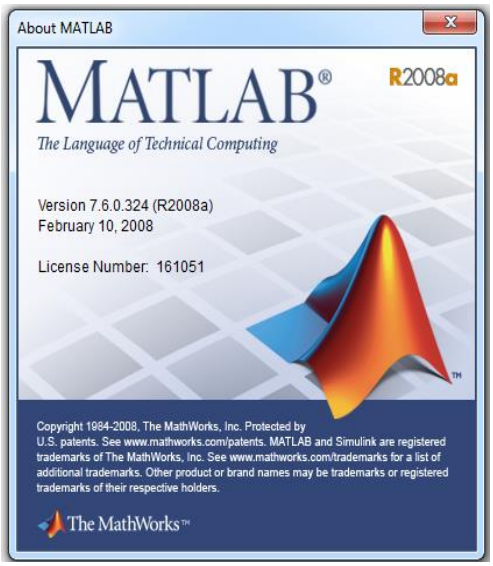
start ⟶ Programs ⟶ MATLAB R2008a

عندها سوف تظهر لنا شاشة تحمل أسم البرنامج MATLAB ونسخة الإصدار وسنة النشر كما في الشكل رقم (1). ثم بعد ثواني قليلة تظهر نافذة البرنامج الرئيسية والتي تكون في بداية

التشغيل كما في الشكل رقم (2) حيث تحتوي هذه النافذة كسائر البرمجيات التي تعمل تحت بيئة نظام Windows على نوافذ فرعية.



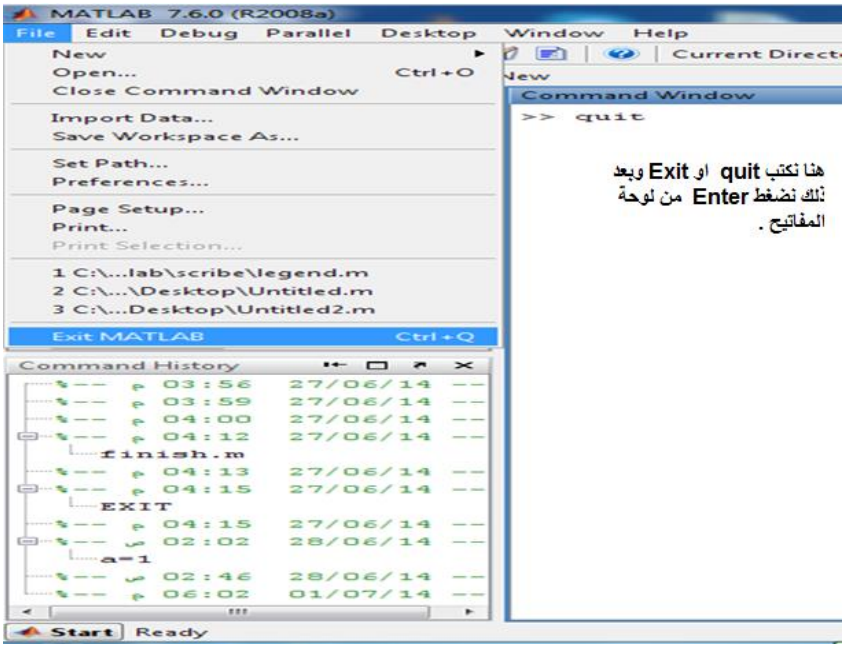
شكل (2): شاشة نافذة البرنامج الرئيسية (سطح مكتب MATLAB)



شكل (1): شاشة اسم البرنامج MATLAB

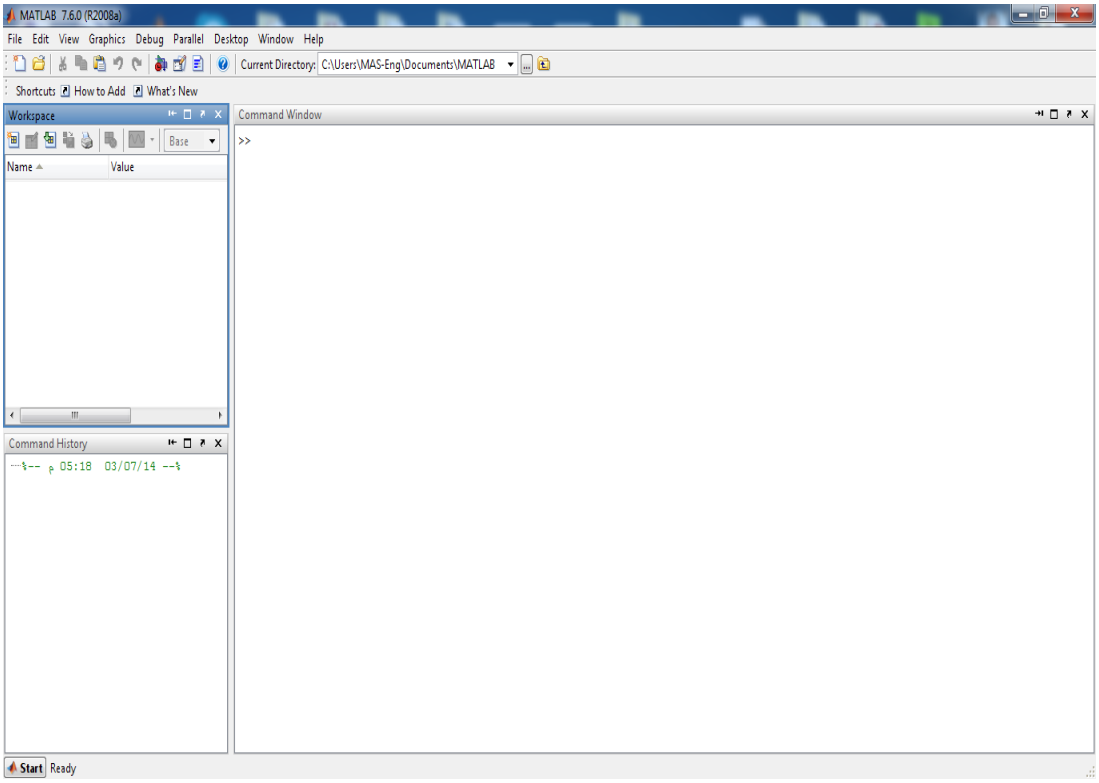
إغلاق: MATLAB

يمكننا الخروج من برنامج MATLAB عن طريق اختيار الأمر (Exit Matlab) من القائمة (File، أو ctrl + Q اوعن طريق كتابة الأمر quit) او exit (في الـ Command Window).



سطح مكتب برنامج MATLAB

عند تشغيل برنامج MATLAB ستظهر على شاشتك عدة نوافذ عنوان احدها MATLAB وتسمى سطح مكتب برنامج MATLAB، تحوي هذه النافذة وتتحكم بجميع النوافذ الأخرى المكونة لبرنامج MATLAB. وحسب خيارات تنصيب البرنامج، فقد تكون بعض هذه النوافذ مرئية أو مخفية ضمن نافذة MATLAB.



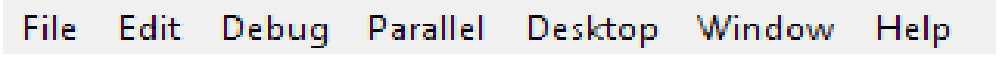
مكونات نافذة MATLAB

تتكون نافذة MATLAB من الأجزاء التالية:-

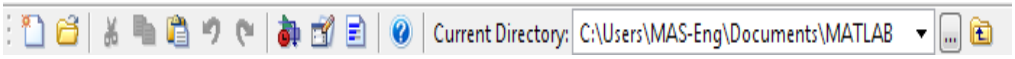
1- شريط العنوان ويكون ذات لون مميز عن باقي الأشرطة يوجد على يساره الرمز الصوري للبرنامج وأسم البرنامج

وفي يمينه

2- شريط قوائم (Menu Bar) أو (Lists Bar) يبدأ بقائمة ملف File، قائمة تحرير Edit، قائمة عرض View، ... وحتى قائمة المساعدة Help.

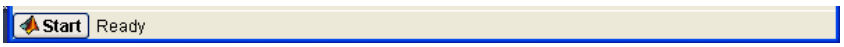


3- شريط الأدوات (Tools Bar) ويضم رموز صورية لبعض الايعازات الموجودة في قوائم الشريط السابق.



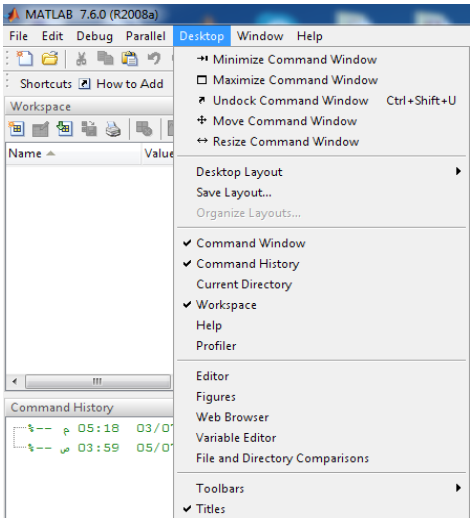
هناك في الجزء الأخير من شريط الأدوات جزء مهم يدعى الدليل الحالي (Current Directory) والذي يخبر المستخدم في أي جزء من الحاسب هو موجود حالياً وكما في الشكل اعلاه يعلمنا بأننا على الدليل (المجلد) MATLAB وعلى القرص C:

4- هنالك شريط مهام خاص بنافذة برنامج MATLAB وفيه كلمتان الأولى Start وعملها كطريق مختصر لتنفيذ بعض الايعازات. بينما Ready تعلمك بأن البرنامج جاهز للعمل حسب التوجيه المعطى له.



بالإضافة إلى الأشرطة أعلاه هناك مجموعة من النوافذ الفرعية التي يمكن تفعيلها أو إخفائها حسب الحاجة وذلك كما في الشكل (3) حيث يتم تأشير أسم النافذة المرغوب بعرضها بإشارة (√)، لكن هناك نافذة أساسية للعمل هي نافذة الأمر Command Window، والتي من خلالها يتم التعامل بكتابة وتنفيذ الأوامر بصورة مباشرة أو غير مباشرة.

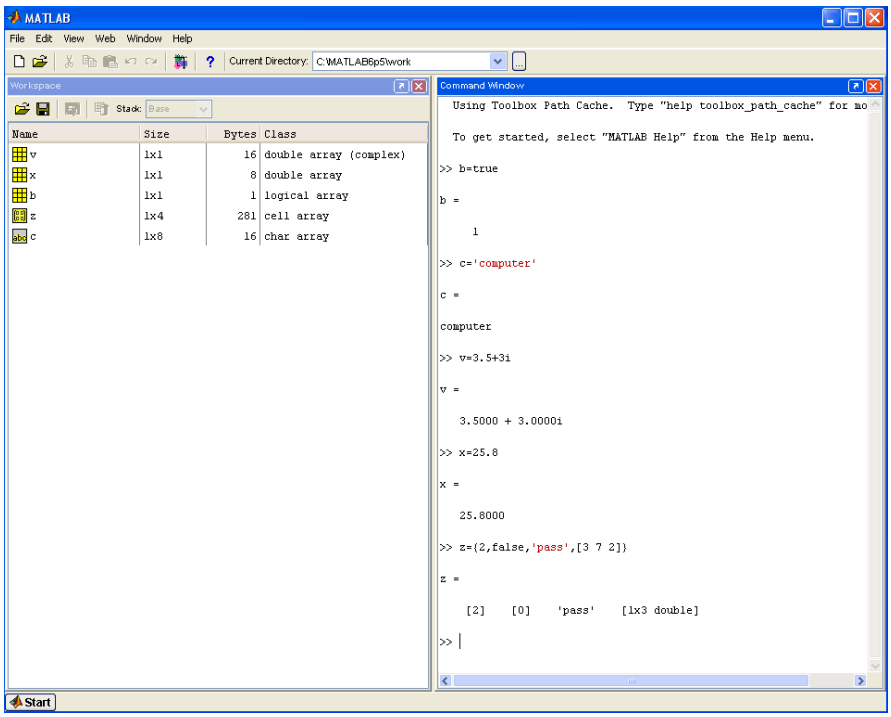
5- تعتبر النوافذ الداخلية الظاهرة أسمائها في قائمة Desktop كما في الشكل رقم (3) هي من مكونات نافذة برنامج MATLAB ولكل نافذة منها عملها الخاص وكما يلي:-



شكل رقم (3) نافذة القائمة Desktop

أ- نافذة الأمر Command Window: وهي نافذة لا يمكن الاستغناء عنها لأن بواسطتها يتم تنفيذ الأوامر وعرض النتائج التي نحصل عليها من تنفيذ تلك الأوامر وتكتب بعد علامة الحث (>>).

ب- نافذة ساحة العمل Workspace: وهي واجهة تخاطبيه تسمح لك باستعراض وتحميل وحفظ متغيرات لغة MATLAB حيث تظهر قائمة تضم أسم المتغير وحجمه وعدد بياناته وصنفه (جميع متغيرات لغة MATLAB هي من صنف مصفوفة)، كما في الشكل (4).

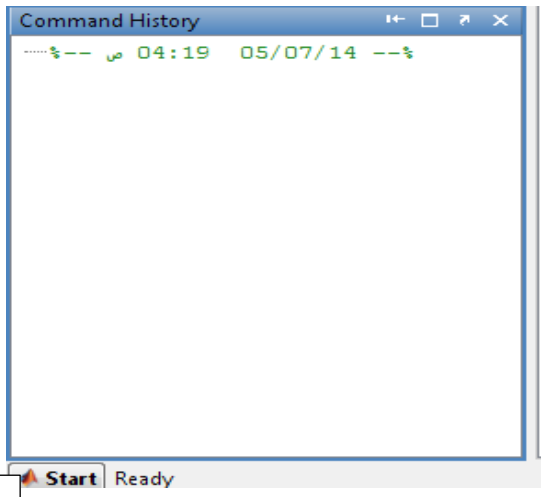


شكل (4): نافذة الأمر Command Window ونافذة ساحة العمل Workspace.

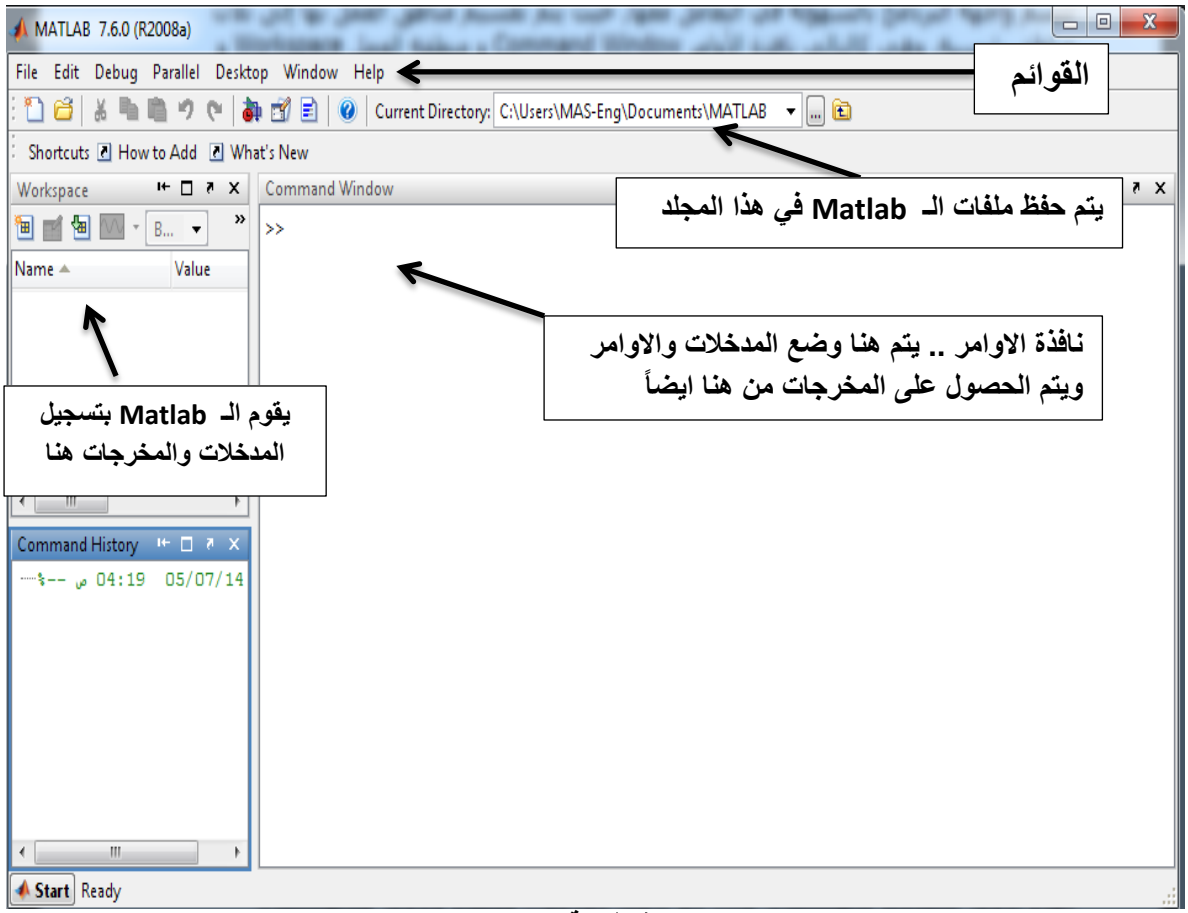
ج- نافذة الدليل الحالي Current Directory: وهي أيضا واجهة رسومية تحدد الدليل الحاوي للملف الذي يتعامل معه برنامج MATLAB.

د- نافذة المساعدة Help: وهي نافذة تخاطبيه (رسومية) تسمح لك بالبحث واستعراض الوثائق بشكل مباشر.

و- نافذة الأوامر السابقة:Command History يتم تسجيل كل ما يقوم به المستخدم على برنامج MATLAB في هذه النافذة وكذلك تمكنك هذه النافذة من إعادة تنفيذ الأوامر السابقة المنفذة في نافذة الأمر بدلاً من كتابتها مرة أخرى.



والشكل (5) يبين النوافذ الداخلية لنافذة البرنامج MATLAB بعد تفعيلها ....

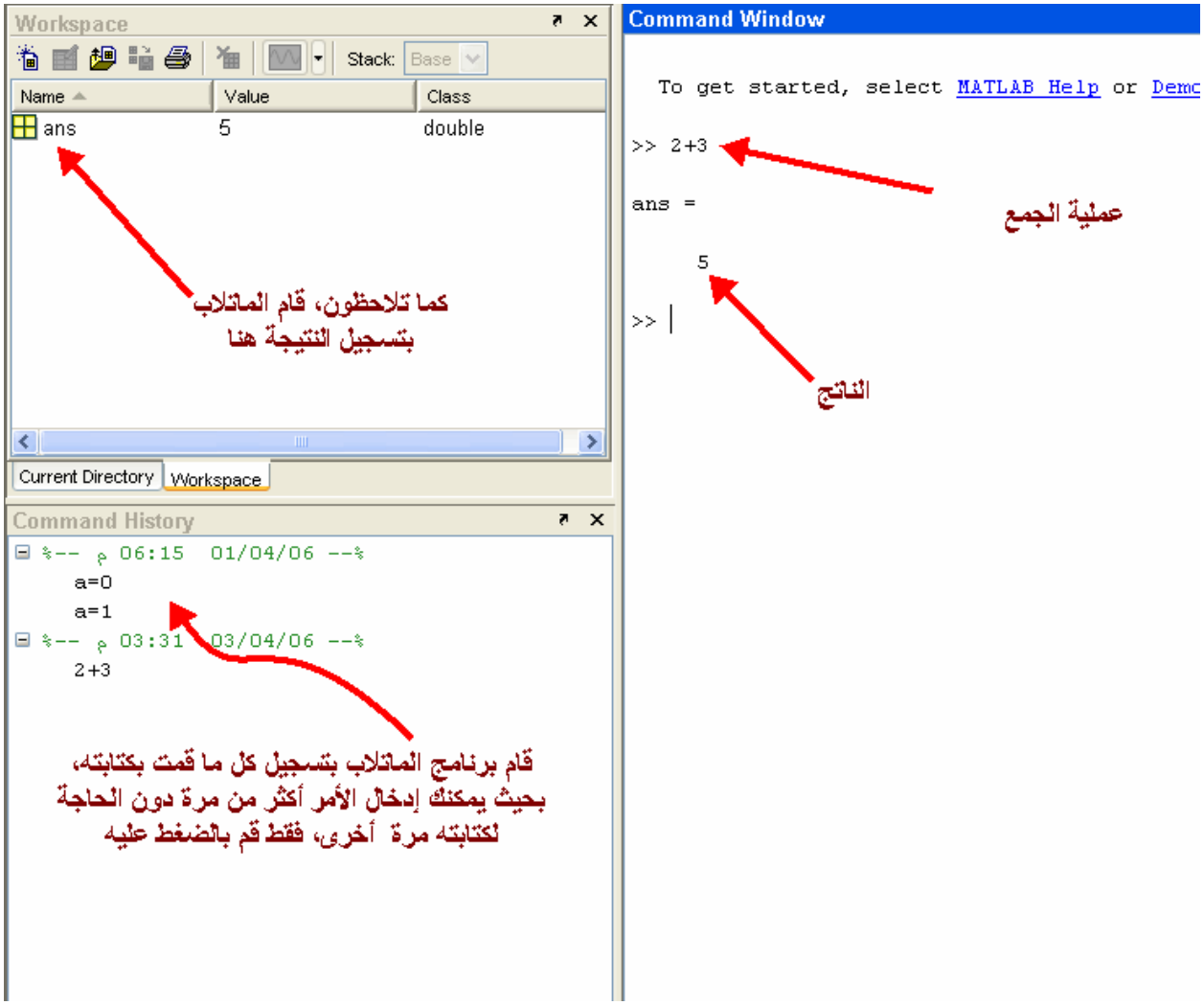


شكل رقم (5)

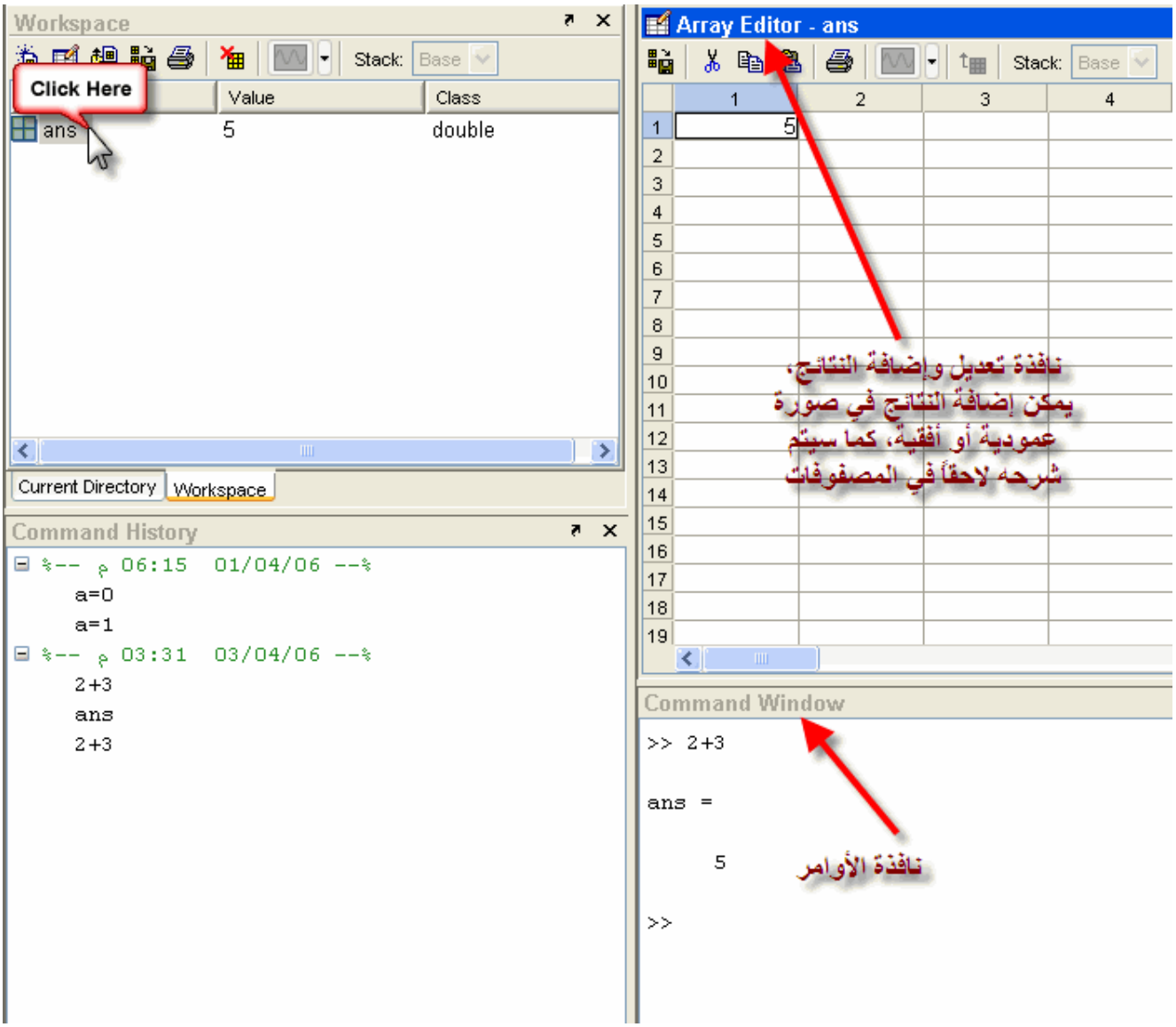
تعريف العمليات الأساسية:

عملية الجمع :

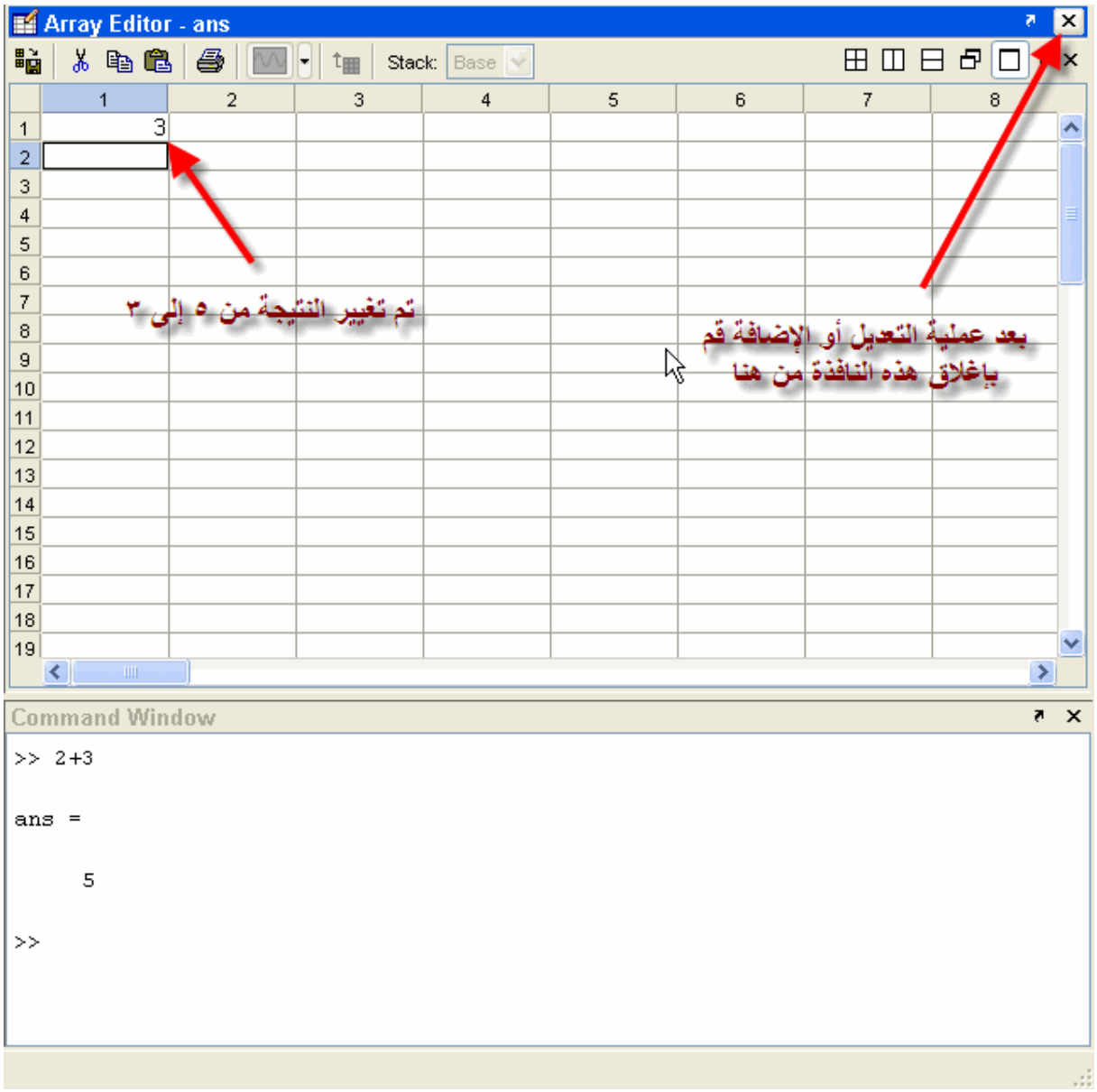
تأخذ علامة الجمع في الماتلاب الرمز المعروف للجمع وهو "+" فمثلاً إذا قمنا بجمع 2+3 سيقوم الماتلاب بوضع الإجابة في صورة أرقام وهو 5, أنظر الصورة التالية:



أذهب إلى نافذة Workspace وقم بالنقر بالماوس بقرة مزدوجة, ستلاحظ ظهور نافذة حلت محل نافذة الأوامر وأصبحت نافذة الأوامر في الأسفل, أنظر الصورة



لنفترض أننا قمنا بتغيير الناتج 5 إلى ,3 قم بإغلاق نافذة تعديل النتائج, كما في الصورة التالية :



ستلاحظ عودة نافذة الأوامر لوضعها الأساسي, قم بكتابة ans في نافذة الأوامر, ستلاحظ ظهور الناتج بالقيمة الجديدة وهي 3, أنظر الصورة :

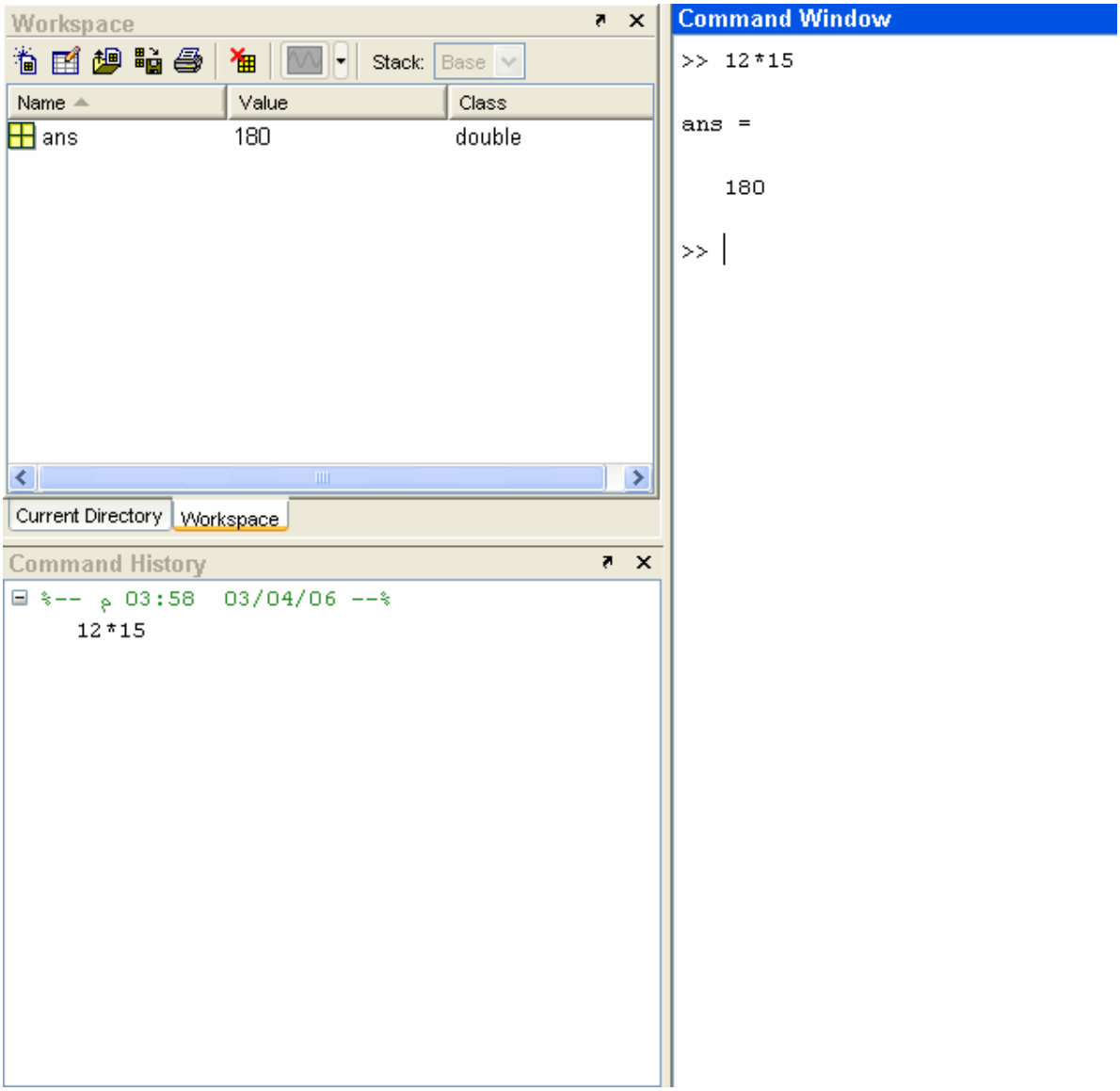
عملية الطرح :

تأخذ عملية الطرح رمز (-) في الماتلاب, فمثلاً 3-2=1, أنظر الصورة

```
>> 3-2

ans =

     1
```

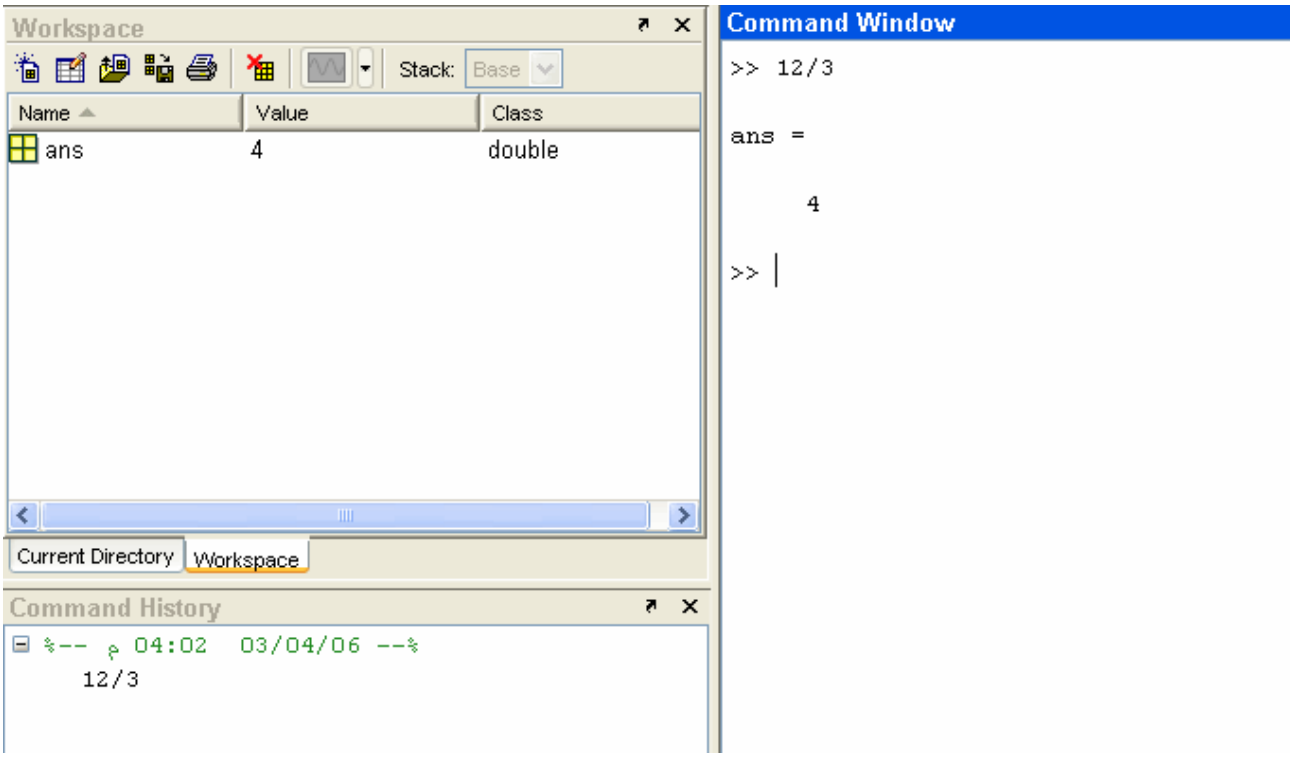
عملية الضرب:

تأخذ عملية الضرب رمز (*), فمثلاً 15*12= 180, أنظر الصورة:



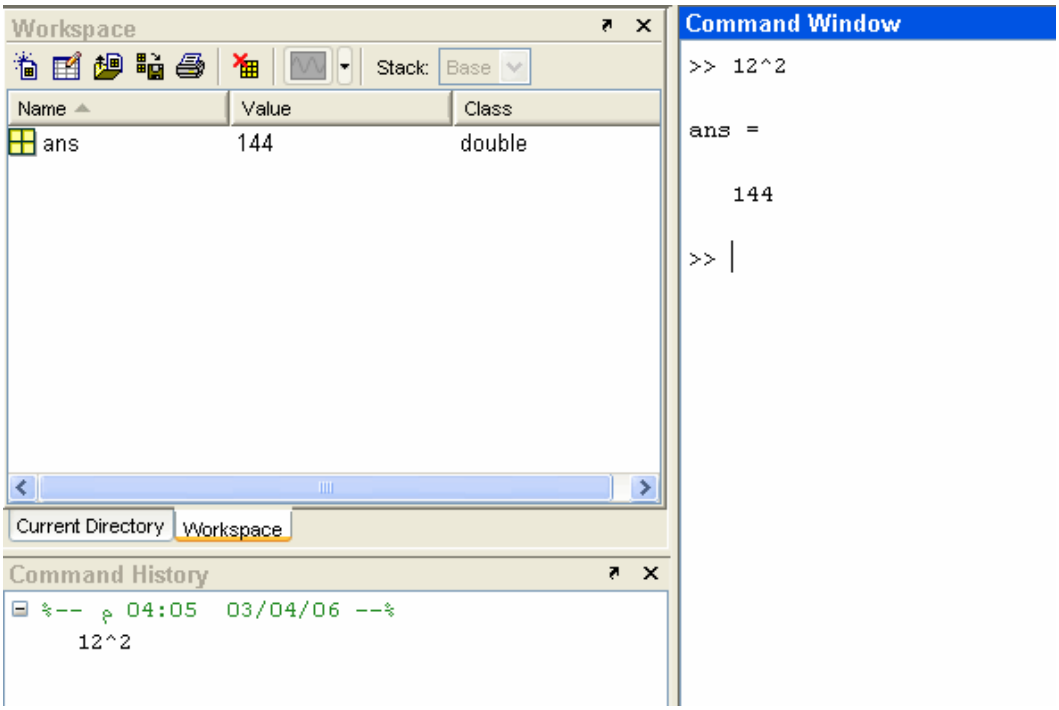
عملية القسمة :

تأخذ عملية القسمة رمز (/), فمثلاً 12 على 3 تساوى 4, أنظر الصورة للتأكد:



عملية وضع الأس:

يأخذ رمز الأس (^), يمكن الحصول على هذا الرمز من خلال الضغط على Shift + 6 في لوحة المفاتيح, فمثلاً 12^2=144, أنظر الصورة :



أخذ الجذر التربيعي :

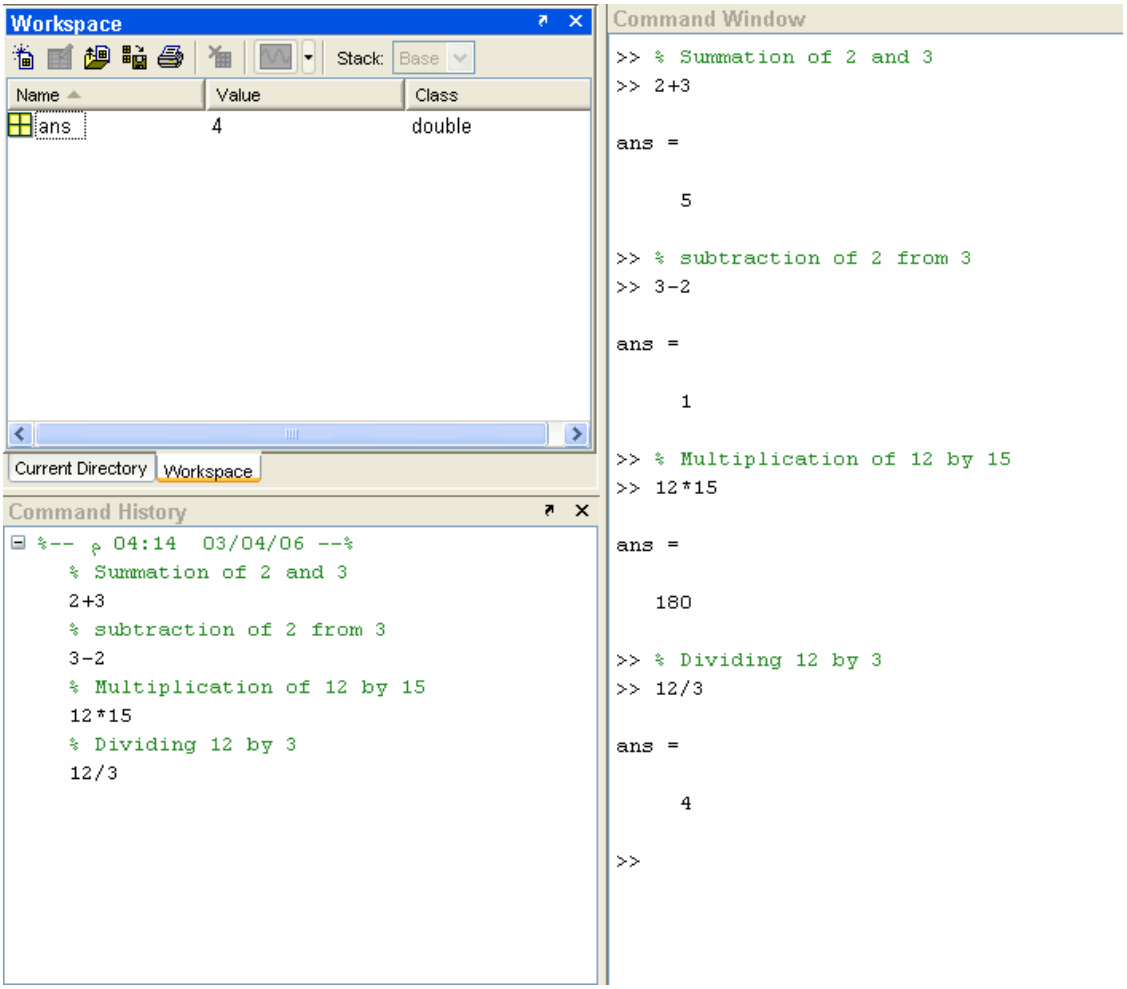
يتم أخذ الجذر التربيعي لأي رقم عن طريق كتابة الأمر ,sqrt فمثلاً الجذر التربيعي للرقم 144 يساوي 12 أنظر الصورة التالية:

```
>> sqrt(144)

ans =

    12
```

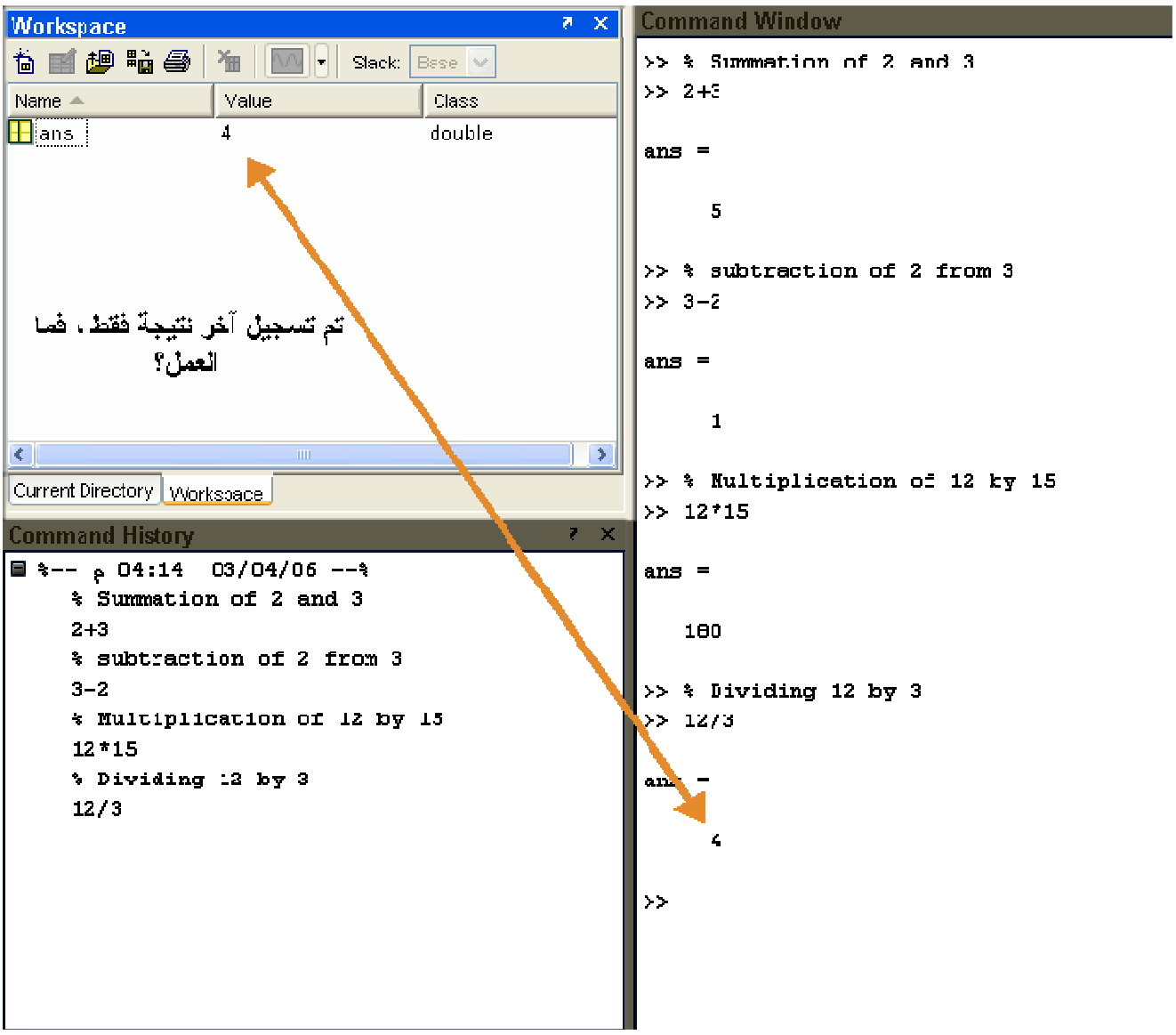
بعض الأوامر الأساسية:

وضع عناوين أثناء البرمجة :

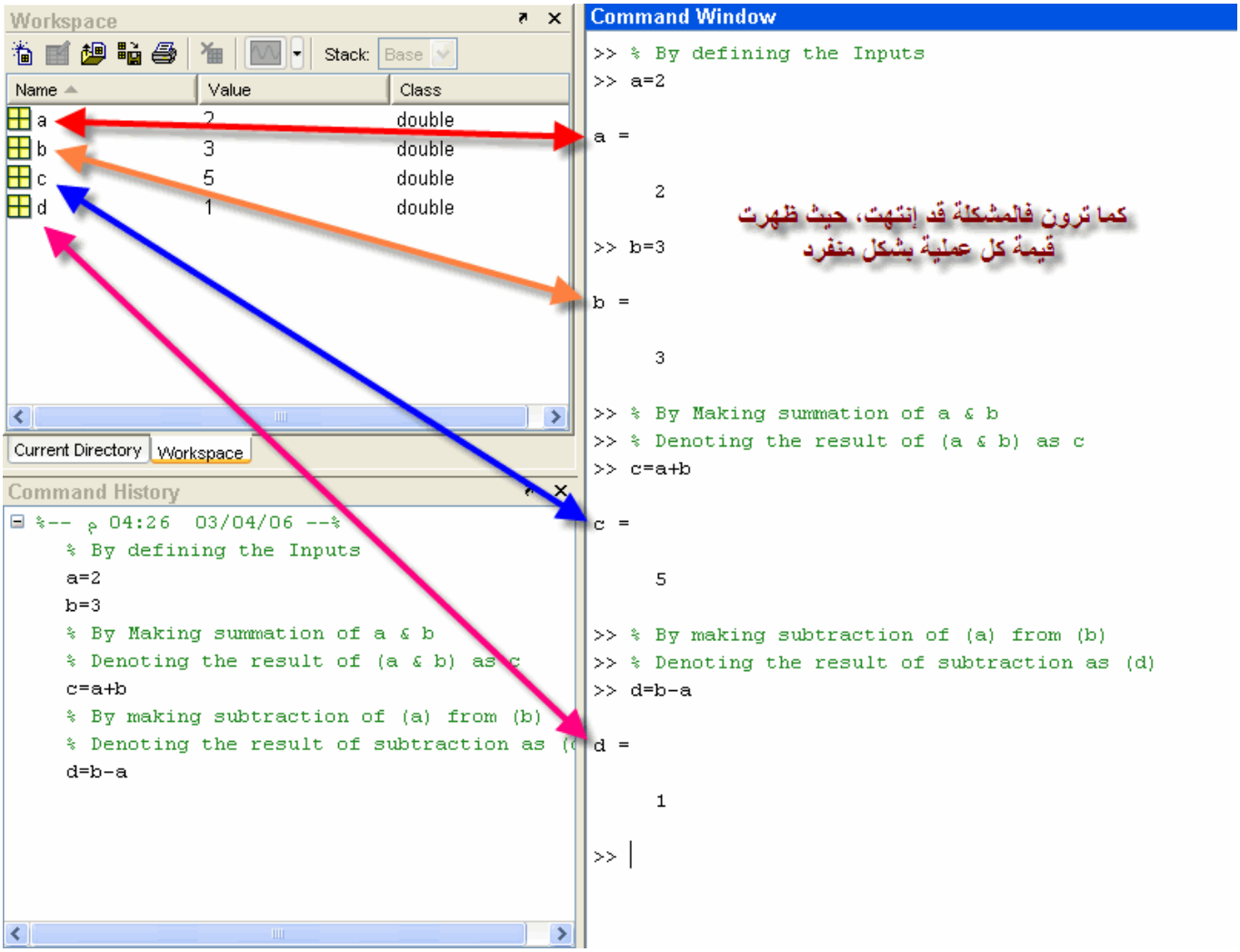
كما تعودنا في برامج Qbasic و C++ وغيرها الكثير من برامج البرمجة, فيتم وضع عناوين لما نقوم به حيث تكون مثل المرجع لنا في معرفة ما نقوم به في جزء ما من البرنامج. ففي برنامج الماتلاب لوضع عنوان ما, لابد من أن نبدأ بوضع علامة مئوية (%), ثم نكتب ما نريده بعدها, لاحظ الصورة التالية :



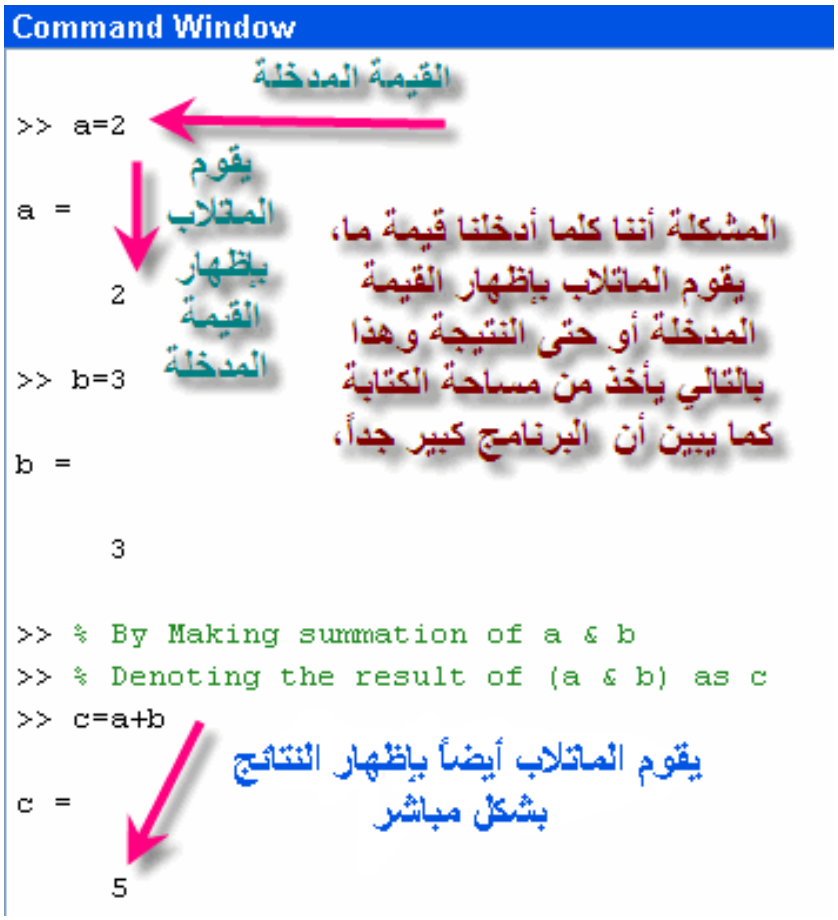
ولكن كما تلاحظون فهنالك مشكلة في نافذة Workspace, حيث أنها سجلت آخر قيمة فقط,
وذلك لأن كل النتائج الأربعة تأخذ رمز ans حيث أننا لم نجع لها رمزاً, أنظر الصورة



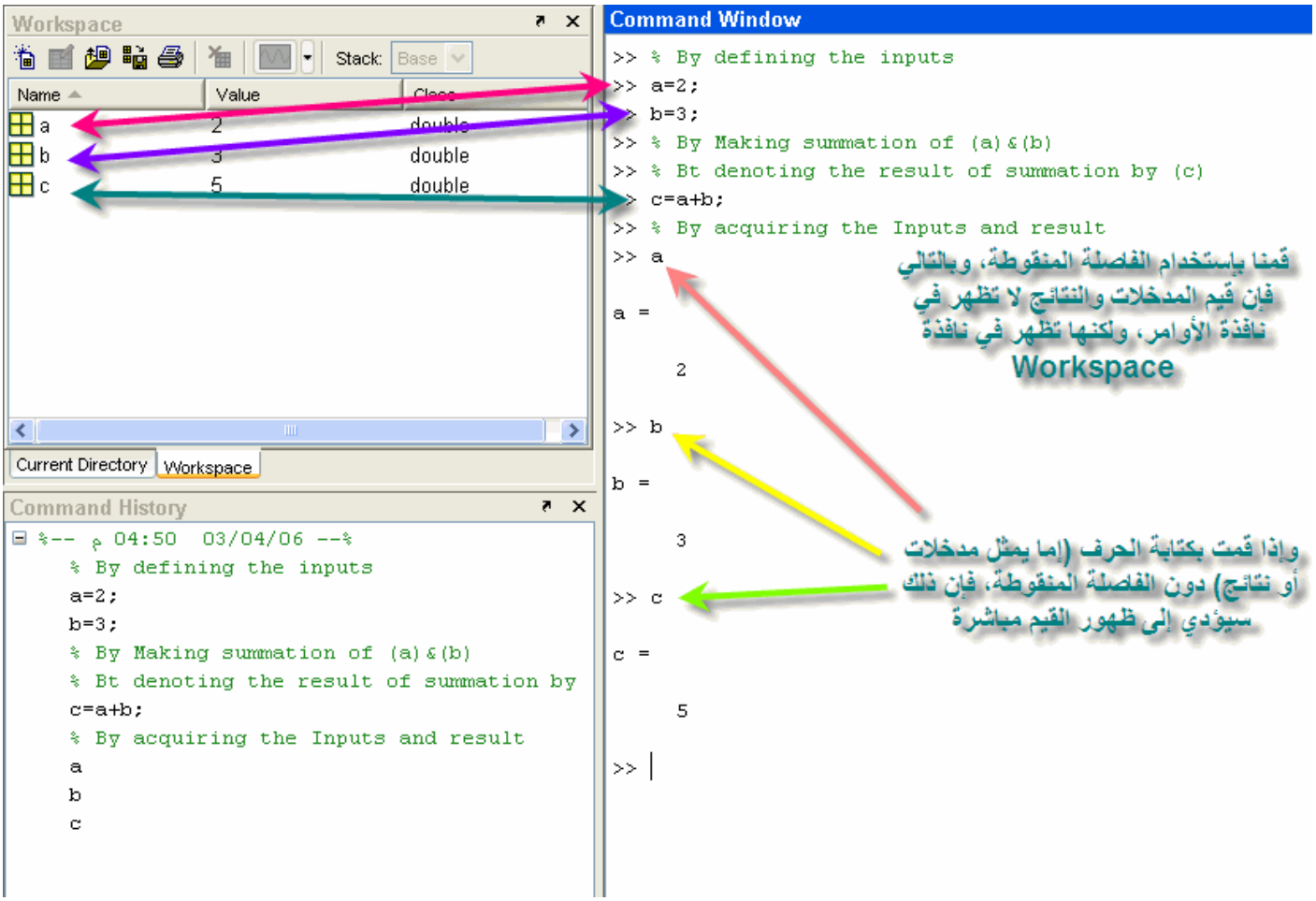
يتم تعريف النتائج بحروف, بحيث يأخذ الحرف القيمة التي يدخلها المستخدم له, أنظر الصورة :



كمـــا تـــرو فالمشـــكلة قـــد انتهـــت تمامـــاً, حيـــث تأخـــذ كـــل قيمـــة حـــر معـــين. المشكلة التالية, هو أننا كلما أ خلنا قيمة, أو حصلنا على نتيجة تكو هنالك مشكلة, هو أ الماتلاب يقوم بإ هار القيمة المدخلة وكذلك النتيجة في نافذة الأوامر, مما يؤ ي إلى كبر البرنامج المكتوب في حين أنه يؤ ي شئ بسيط, أنظر الصورة التالية :

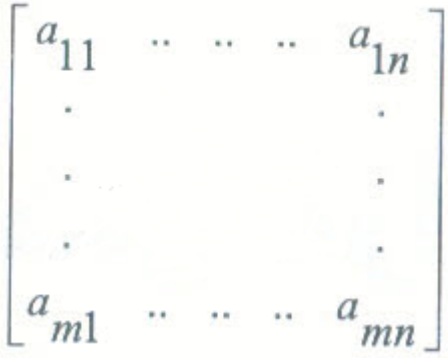


يتم أخفاء القيمة المدخلة وكذلك النتيجة من الظهور (ولكن عملية إدخال النتيجة والجمع مثلاً تتم بشكل طبيعي ويقوم الماتلاب بتنفيذ ما يأمره المستخدم) عن طريق وضع علامة (;) بعد كل قيمة مدخلة أو بعد طلب نتيجة ما (الجمع مثلاً) ويتم إظهار النتيجة أو القيم المدخلة إذا طلب المستخدم ذلك , عن طريق وضع حرف المدخلات أو النتيجة المطلوبة دون استخدام الرمز الرمز المذكور (;) انظر الصورة التالية :


Workspace
Stack: Base
Name
Value
Class
a
2
double
b
3
double
c
5
double
Current Directory
Workspace
Command History
%-- م 04:50 03/04/06 --%
% By defining the inputs
a=2;
b=3;
% By Making summation of (a)&(b)
% Bt denoting the result of summation by
c=a+b;
% By acquiring the Inputs and result
a
b
c
Command Window
>> % By defining the inputs
>> a=2;
>> b=3;
>> % By Making summation of (a)&(b)
>> % Bt denoting the result of summation by (c)
>> c=a+b;
>> % By acquiring the Inputs and result
>> a
a =
2
>> b
b =
3
>> c
c =
5
>>
قمنا بإستخدام الفاصلة المنقوطة، وبالتالي فإن قيم المدخلات والنتائج لا تظهر في نافذة الأوامر، ولكنها تظهر في نافذة Workspace
وإذا قمت بكتابة الحرف (إما يمثل مدخلات أو نتائج) دون الفاصلة المنقوطة، فإن ذلك سيؤدي إلى ظهور القيم مباشرة

ما هي المصفوفة Matrix:

المصفوفة Matrix هي مجموعة من الارقام (القيم العددية) توضع في صورة صفوف واعمدة وتستخدم في ايجاد حلول من المسائل التطبيقية التي تواجهنا في دراستنا ويتم استخدامها في ادخال البيانات والحصول على المخرجات ،كما موضح بالصورة :



حيث m تمثل عدد الصفوف و n تمثل عدد الاعمدة .

تعريف المتجهات Vector

هي مجموعة من الارقام توضع في صورة صف واحد وتسمى في هذه الحالة متجهات صفية Row Vectors او عمود واحد وتسمى متجهات عمودية Column Vectors وبالتالي فهي تمثل مصفوفة احادية .

ما الفرق بين المتجهات والمصفوفات؟

المتجهات هي مصفوفة ولكن إما بعمود واحد Column Vector أو صف واحدRow Vector

المتجه هو عبارة عن مجموعة من الارقام توضع في صورة صف واحد او عمود واحد لذا فان المتجهات حالة خاصة من المصفوفات حيث ان المصفوفات يزيد عدد صفوفها واعمدتها عن صف واحد وعمود واحد .

ملاحظات:

1- كل متغير في MATLAB عبارة عن مصفوفة، لغة MATLAB موجهة بالمصفوفات حتى وان كانت المتغيرات أعداداً مفردة (scalar).
2- الأمر clear ضمن Workspace يستخدم لحذف المتغيرات والدوال من الذاكرة .

3- الأمـر clc ضـمن Workspace يسـتخدم لمسـح نافـذة الأمـر Window Command.

4- ثـلاث نقـاط متتاليـة (...) فـي نهايـة السـطر مسـبوقة بفـراغ للدلالـة علـى اسـتمرار الايعاز في السطر التالي.

5- فـارزة منقوطـة بعد الإيعـاز تمنـع طباعـة المتغير أو النـاتج فـي نافذة Command وكذلك في نافذة Editor.

6- إشارة النسبة المئوية (%) تستخدم للتعليق فكل نص يأتي بعدها يعتبر نـص تعليق، مثل:

% This Program Compute Area

7- ملفات MATLAB تسمى M–files وتكون توسعها (.m)، مثلاً (examplel.m).

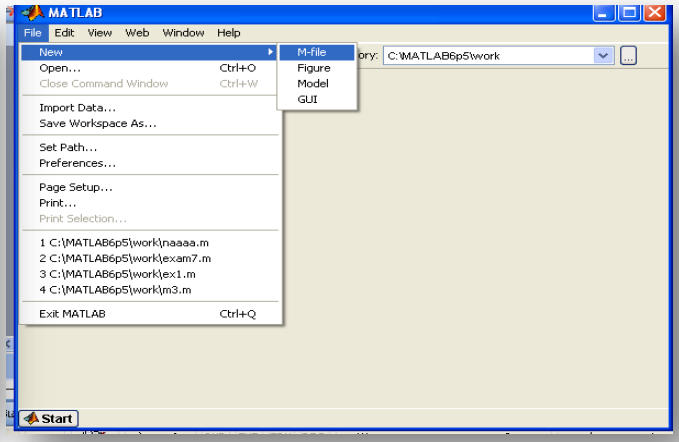
8- الاحتفاظ بكتابـة الايعـازات السـابقة واللاحقـة فـي نافذة Command بحركـة السـهم للأعلى والأسفل.

9- نتيجـة تنفيـذ برنـامج MATLAB (النتـائج والاخراجـات) تظهـر فـي شاشـة Command Window لذلك يجب الانتقال إليها بعد التنفيذ.

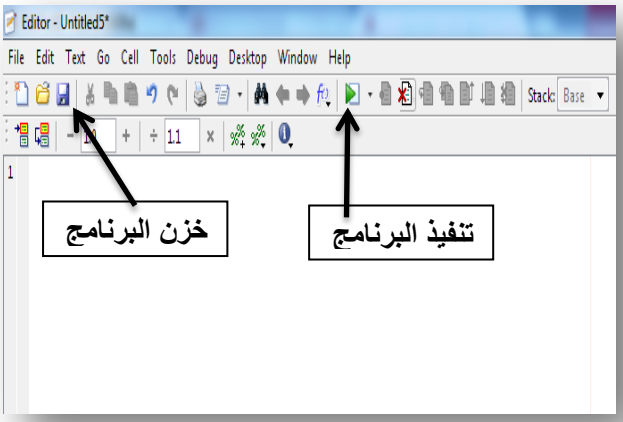
10- لإنشـاء ملـف نصـي M-file، أنقـر علـى أيقونـة الصـفحة الفارغـة (البيضاء) الموجودة ضمن شريط أدوات سطح مكتب MATLAB، أو اختر New من القائمة File ومن ثم اختر M-file أو اختيار Open لفتح ملف موجود مسبقا. يسـتدعي هـذه الأوامـر نافـذة محـرر النصـوص التـي يمكنـك فـي كتابـة أوامـر MATLAB (نافذة كتابة البرامج). كما في الأشكال (6)، (7)، (8).

11- يمكن تنفيذ الملف المخزون باختيـار أيقونـة Run الموجودة في شـريط أدوات نافذة Editor أو عبر ضغط المفتاح F5 أو الاختيار Run من القائمة Debug، أو كتابـة اسم الملـف المخـزون أمـام علامـة الحـث << فـي نافذة Command. بعد انتهاء كتابة البرنامج (الملف) يخزن هذا الملف كملف M-file باسم معين (مثلاً example1.m) على قرصك الصلب عبر اختيار الاختيار Save من القائمة File أو الخزن ضمن شريط أدوات سطح مكتب MATLAB). كما في الشكل (9).

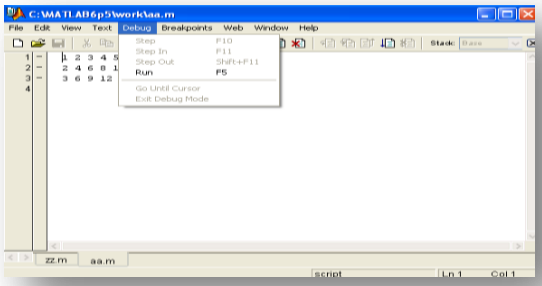
12- عند الخزن يحمل البرنامج اسم افتراضي (Untitle1) فبالإمكـان إعطـاء الاسم الذي يرغب فيه المبرمج أو البقاء عليه.



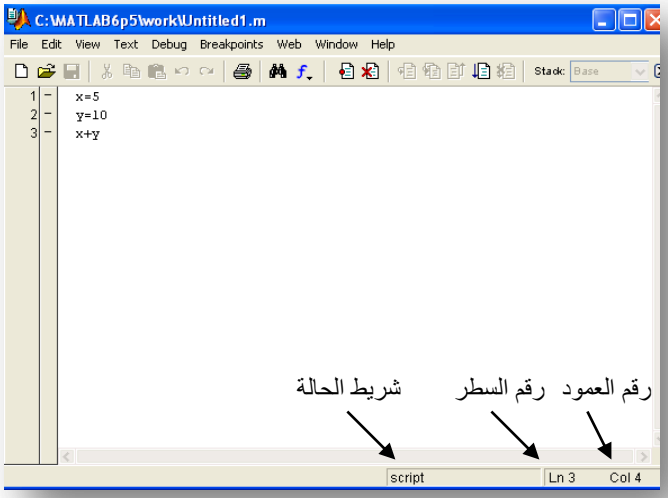
شكل (6): إنشاء ملف جديد.



شكل (7): نافذة كتابة البرامج (محرر الملفات النصية).



شكل (9): تنفيذ البرنامج.



شكل (8): نافذة كتابة البرامج (محرر الملفات النصية).

13- عندما نريد إغلاق برنامج MATLAB عبر الاختيار Exit MATLAB من القائمة File الموجودة في نافذة سطح مكتب MATLAB أو عبر كتابة الأمر Exit في نافذة Command، أو علامة (×) في زاوية سطح مكتب MATLAB العليا اليمنى.

رموز لغة MATLAB : MATLAB Symbols

تتكون لغة MATLAB من العناصر الأساسية التالية:

أ- حروف أبجدية إنكليزية: وهي: A, B, … , Z, a, b, … , z

ب- أرقام حسابية: 0, 1, 2, …, 9

ج- رموز خاصة مثل:) , (, * , ; , < , > , =, - , + ,{ , … الخ.

الثوابت Constants:

يوجد في لغة MATLAB أنواع متعددة من الثوابت أهمها:-

(أ) الثوابت العددية Numerical Constants:

وتتكون من عدد من الأرقام ولها عدة أشكال هي:

(1) **الثوابت الصحيحة Correct Constants:**

مثل: 0, +23, 472, -18

ملاحظة: أكبر عدد صحيح مستخدم.

>> bitmax or >>flintmax (في الإصدارات الاحدث)

ans =

9.007199254740991e+015

والتي تقابل $2^{53}-1$

(2) **الثوابت الحقيقية Real Constants:**

مثل: 0.0, 51.8, 472.5, -18.0

ملاحظة:

>> realmin

ans =

```
  2.225073858507201e-308

>> realmax

ans =

  1.797693134862316e+308

>> pi

ans =

  3.146
```

(ب)الثوابت المنطقية Boolean Constants:

وهي الثوابت التي قيمتها العددية (1) في حالة true و (0) في حالة false.

مثال:

3 > 2 ⟹ 1

0 > 5 ⟹ 0

المتغيرات Variables:

هناك بعض القواعد الواجب مراعاتها عند كتابة اسم المتغير وهي:

1. لا يمكن استخدام الكلمات المفتاحية (الكلمات المحجوزة) أو الدوال التي توفرها اللغة كأسماء متغيرات، مثال:

   if, end, for, break, else, global, return, function, sin, log, …

2. أسماء المتغيرات حساسة لحالة الحرف (COST, CoST, cost, Cost متغيرات مختلفة، وكذلك A و a).

3. حرف l (small letter) في لغة MATLAB يشبه رقم 1.

4. يمكن لأسماء المتغيرات أن تحوي 63 رمزا وسيهمل أي رمز زائد عن 63.

5. يجب أن تبدأ أسماء المتغيرات بحرف متبوعا بأي عدد من الأرقام أو الأحرف أو underscore. ولا يجوز استخدام الرموز الخاصة أو الفراغ.

6. جميع أوامر MATLAB تكتب بالحروف الصغيرة (if, while, input, ...).

هناك عدة أنواع من المتغيرات في لغة MATLAB وهي:

(أ) المتغيرات العددية Numerical Variables:

تتكون من حرف واحد أو مجموعة من الحروف من A إلى Z و a إلى z ويمكن أن يحتوي على أرقام من 0 إلى 9 ويمكن أن تكون سلسلة من الأرقام والحروف بشرط أن يبدأ بحرف (خليط من أرقام وحروف مبدوءة بحرف) ويمكن كذلك أن يحتوي المتغير على underscore حتى 63 رمزاً. وتكون قيمة المتغير عددية (صحيح، حقيقي، عقدي أو أسي).

مثال:

Ali_Ahmed, X2, S2, ks, K

التعبير الحسابي Arithmetic expression

يتكون التعبير الحسابي من مجموعة من الثوابت والمتغيرات تجمع بينهما عمليات حسابية ويستخدم فيها الرموز الحسابية مثل +، -، /، *، ^ والأمثلة الآتية تعبر عن تعابير جبرية صيغت بلغة MATLAB.

التعبير الجبري	التعبير بلغة MATLAB
$a - 3b$	a – 3 * b
$c^2 - 10$	c ^ 2 – 10
$a^2 + b^2 / 12$	(a ^ 2 + b ^ 2) / 12
$m (7d - 8g)$	m * (7 * d – 8 * g)

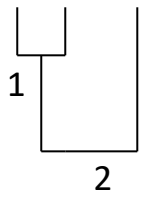
قاعدة الأسبقية (الأولوية) Rule of Precedence

وهذه القاعدة مهمة في فهم وترتيب أولويات العمليات الحسابية في التعابير والمعاملات الحسابية، كما يجريها وينفذها الحاسب، وتنص القاعدة على أن الأولوية الأولى تعطى للعمليات الموجودة بين القوسين ومن اليسار إلى اليمين، وبالنسبة للعمليات الحسابية فالرفع إلى الأس أولا، والضرب (أو القسمة) ثانيا، والجمع (أو الطرح) أخيراً والمثال التالي يوضع هذه القاعدة:

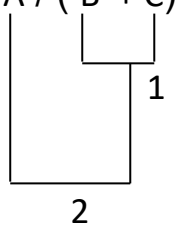
انظر الجدول الذي يبين تعبير العملية الحسابية بواسطة الماتلاب وأولوياتها

Symbol	Operation	Matlab Form
( )	الاقواس	(AB)
^	القوه او الاس	A^B
*****	الضرب	A*B
/	القسمة	A/B
+	الجمع	A+B
-	الطرح	A-B

التعبير: A / B + C يكافئ في الجبر $\frac{A}{B}+C$

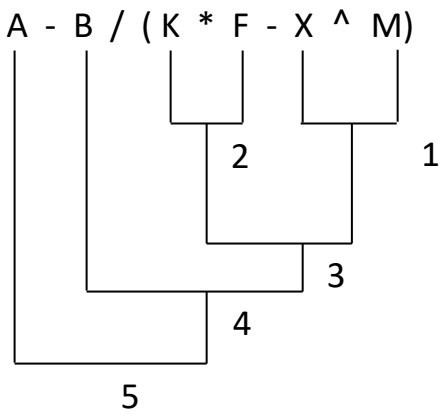


بينما يكافئ التعبير A / (B + C) يكافئ في الجبر $\frac{A}{B+C}$



لان الجمع داخل الأقواس يجري أولاً حسب الأولوية ثم يقسم A على نتيجة القوس.

مثال: التعبير



تنفيذ العمليات حسب الخطوات التالية:

تأخذ الأقواس الأولوية الأولى، وتنفذ العمليات داخلها حسب الأولوية أيضا.

العملية الأولى: رفع X إلى الأس M لتصبح كمية واحدة.

العملية الثانية: ضرب K في F لتصبح كمية واحدة.

العملية الثالثة: طرح نتيجة العملية الأولى من نتيجة العملية الثانية وتصبح النتيجة كمية واحدة.

العملية الرابعة: تقسم B على نتيجة العملية الثالثة وتصبح النتيجة كمية واحدة.

العملية الخامسة: تطرح نتيجة العملية الرابعة من A وتصبح النتيجة كمية واحدة.

الجملة الحسابية Arithmetic Statement

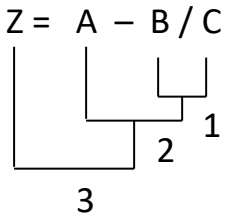
الجملة الحسابية في MATLAB تكافئ المعادلة الحسابية في الجبر إلا أن MATLAB تشترط أن يكون اسم المتغير المراد حساب قيمته في الطرف الأيسر وحده بدون أشاره بينما يكون التعبير الحسابي (بقية المعادلة) في الطرف الأيمن، كما في الأمثلة التالية:

1) y = A * X + B

2) A = 3.14 * R ^ 2

مثال:

أولوية العمليات الحسابية في الجمل الحسابية:



يمكن ملاحظة أن إشارة المساواة تمثل آخر أولوية حسابية بعد انتهاء جميع العمليات الحسابية في الطرف الأيمن.

(ب) المتغيرات الرمزية String Variables:

تشبه في تركيبها المتغيرات العددية والفرق الوحيد بينهما هو أن قيمة المتغير الرمزي تكون رمزية (مكتوبه بين علامتي اقتباس).

الجملة الرمزية String Statement

تشبه في تركيبها الجملة الحسابية والفرق الوحيد بينهما هو أن المتغير في طرفها الأيمن يكون رمزياً (مكتوبه بين علامتي اقتباس) والتعبير في طرفها الأيسر يكون متغير.

والأمثلة التالية توضح ذلك:

A = 'Ali muhammed ali '

N = 'Number of Student'

Dept ='Computer Science';

ملاحظة: التعابير في الطرف الأيمن لا يكون لها قيم حسابية لو استخدمت في عمليات حسابية لأنها موضوعة داخل ' '.

الاقترانات المكتبية Library Functions:

يتوفر في معظم الحاسبات باستخدام لغة MATLAB اقترانات رياضية يكثر استعمالنا لها، مثل الدوال والاقترانات المثلثية واللوغارتيمية وغيرها ويمكن استدعائها في أي وقت، ومنها:

الاقتران	المعـــــنى
Sqrt	الجذر التربيعي
abs	القيمة المطلقة
exp	المرفوع إلى قوة بأساس 10
log	اللوغاريتم الطبيعي
$\log_{10}$	اللوغاريتم العشري
$\log_{2}$	اللوغاريتم ذو الأساس 2
sin	جيب الزاوية
Cos	جيب تمام الزاوية
Tan	ظل الزاوية
atan	ظل معكوس الزاوية
fix	التدوير باتجاه الصفر

floor	التدوير باتجاه اللانهاية السالبة
ceil	التدوير باتجاه اللانهاية الموجبة
round	التدوير باتجاه أقرب عدد صحيح
mod	الجزء الصحيح من حاصل القسمة
rem	بقية القسمة
Sign	إشارة العدد إذا كانت موجبة، سالبة، صفر
imag	القسم التخيلي
real	القسم الحقيقي
factor	العوامل الأولية
Isprime	يعيد true إذا كان العدد أوليا
primes	ينشئ قائمة بالأعداد الأولية
gcd	القاسم المشترك الأعظم
lcm	المضاعف المشترك الأصغر

مثال:

>> x=3.5;

m1=sqrt(3.5) ; m2=sin(x); m3=fix(x); m4=round(x);

m1 = 1.87.8

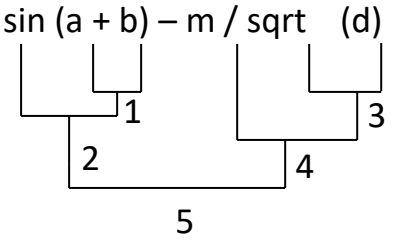
m2 = -0.3508

m3 = 3

m4 = 4

ملاحظة:

تأخذ الاقترانات المكتبية أولوية بعد الأقواس عند تنفيذ العمليات الحسابية.



يكون تنفيذ العمليات الحسابية كما يلي:

العملية الأولى: إيجاد قيمة جمع a مع b.

العملية الثانية: إيجاد قيمة جيب الزاوية لناتج العملية (1).

العملية الثالثة: إيجاد قيمة الجذر التربيعي لـ d.

العملية الرابعة: إيجاد ناتج قيمة ناتج قسمة m على ناتج العملية (3).

العملية الخامسة: طرح ناتج العملية (4) من ناتج العملية (2) وتصبح النتيجة النهائية كمية واحدة (عدداً واحداً).

مثال: تمثل الجمل التالية إقترانات مكتبية في الجبر وقيمتها في MATLAB:

b = sqrt (a ^ 2 + 10) $\Longleftarrow$ $b = \sqrt{a^2 + 10}$

z = log (c * x + n * y) $\Longleftarrow$ z = ln (cx + ny)

y = (sin (x + n * k)) ^ 3 $\Longleftarrow$ $y = \sin^3 (x + nk)$

s = atan (y / x) ⇐ s = $\tan^{-1}$ (y / x)

r = 2 * sqrt (exp (x – 5)) ⇐ $r = 2\sqrt{e^{x-5}}$

t = abs (x – sqrt (y)) / (a + m) ⇐ $t = \left|x - \sqrt{y}\right| \Big/ (a + m)$

g = p ^ (3 / 2) + (a * b / c) ^ (1 / 5) ⇐ $g = p^{3/2} + \sqrt[5]{ab/c}$

المصفوفات والعمليات عليها

لقد كانت جميع العمليات التي أجريتها حتى الآن مؤلفة من أعداد وحيدة البعد سنسميها أعداد مفردة. وتعتبر العمليات على الأعداد المفردة هي أساسيات علم الرياضيات. وبنفس الوقت، وعندما يريد الشخص إجراء نفس العملية على عدد مفرد أو أكثر، فسيحتاج إلى إعادة إجراء العملية عدة مرات، مما يعني هدر في الوقت والجهد. ولحل هذه المشكلة، عمد برنامج MATLAB إلى إجراء العمليات الرياضية على مصفوفة من البيانات.

المصفوفة البسيطة

يتعامل برنامج MATLAB مع المصفوفات بشكل مباشر وبطريقة سلسة، إذ أن إنشاء المصفوفات يتم بطريقة سهلة جداً.

مثال (١) :

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

p =

   1   2   3   4   5   6   7   8   9   10   11   12
```

مثال (٢) :

```
y=[ 1 ,2 ,3  ; 4 , 5 , 6 ; 7 , 8 , 9 ]

y =

   1   2   3

   4   5   6

   7   8   9
```

أو

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

y =

   1   2   3

   4   5   6

   7   8   9
```

يقتصر كل ما عليك لإنشاء مصفوفة في لغة MATLAB على أن تبدأ بقوس يساري ثم تدخل القيم المطلوبة بفراغ أو (فارزة) ثم أغلق المصفوفة بقوس يميني. وعندما تريد كتابة y او p فأن برنامج MATLAB يعلم بأنك تريد حساب قيم لكل من الـ y , p ويقوم بوضع النتائج في مصفوفة أخرى مما يجعل امكانية MATLAB مختلفة عن لغات البرمجة الأخرى.

عنونة المصفوفة أو الفهرسة

المصفوفات أعلاه تتكون من ٩ و ١٢ عنصر، يمكن الوصول إلى أي عنصر منها باستخدام الفهرسة له.

```
>> y (2)

ans =

 4

>> y (7)

ans =

    3
```

ولتعريف مجموعة من العناصر بنفس الوقت فأن برنامج MATLAB يستخدم النقطتين المتعامدتين (:).

```
>> y (1: 5)

ans =

1    4    7    2    5
```

هذه هي العناصر الخمسة الأولى من المصفوفة y، ويخبرك الرمز 1: 5 بأن تبدأ بالرقم 1 وتعدّ حتى الرقم 5.

مثال:

```
>> y (7: end)

ans =

   3    6    9
```

وهنا تكمل من العنصر السابع وحتى نهاية المصفوفة، إذ تشير الكلمة end إلى آخر عنصر من عناصر المصفوفة.

مثال:

```
y  (4: -1: 1)

ans =

   2    7    4    1
```

هنا العنصر الرابع ثم الثالث ثم الثاني ثم الاول بترتيب عكسي، ويخبرك الرمز 4: -1: 1 بأن تبدأ بالرقم 4 وتعدّ نزولاً بقيمة 1 وتقف عند الرقم 1.

مثال:

```
p (2: 2: 7)

ans =

   2    4    6
```

هنا العنصر الثاني والرابع والسادس من المصفوفة p، ويخبرك الرمز 2: 2: 7 بأن تبدأ بالرقم 2 وتعدّ نحو الأعلى بـ 2 وتقف عندما تصل إلى الرقم 7.

مثال:

```
>> y ([8  2  7  6])

ans =

6   4   3   8
```

استخدمنا هنا مصفوفة أخرى [6 7 2 8] لوضع عناصر المصفوفة y بالترتيب الذي نرغب فيه، حيث وضع العنصر الثامن أولاً والعنصر الثاني ثانيا، بينما وضع العنصر السابع ثالثاً والعنصر السادس رابعا. في الواقع تدل المصفوفة [6 7 2 8] عناوين العناصر المرغوبة من المصفوفة y.

مثال:

```
>> y ([1   1   3  4   2   2])

ans =

1   1   7   2   4   4
```

إنشاء المصفوفة

لقد قمنا سابقاً بإدخال قيم مصفوفة y عبر كتابة كل العناصر ضمن المصفوفة، وهنا الأمر مقبول لان المصفوفة y تحتوي على تسعة عناصرً فقط، ماذا لو احتوت 111 عنصراً؟

هناك عدة طرق لإدخال عناصر المصفوفة y .

أمثلة:

1) `>> x = (0: 0.1: 1) * pi`

المتغير

2) `>> x = linspace (0, pi, 11 )`

القيمة الابتدائية القيمة النهائية عدد القيم

3) >> x = [1 2 3 4 5 6 7 8]

وهنالك طرق اخرى سوف يتم التطرق اليها لاحقاً .

مثال:

```
>> a = [1: 7]

a =

   1   2   3   4   5   6   7
```

مثال:

```
>> b = [linspace (1, 7, 5)]

b =

   1   2.5   4   5.5   7
```

مثال:

```
>> (1:6)

ans =

   1   2   3   4   5   6
```

ملاحظة : نستطيع ان ننشأ مصفوفة بدون متغير كما في المثال اعلاه ، وكلمة ans هي اختصار لكلمة Answer فاصبح المتغير الذي يمثل المصفوفة هو ans .

مثال :

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

A =

   1   2   3

   4   5   6

   7   8   9
```

مثال:

```
>>  a = 1: 5    ,   b = 1: 9

a =

   1   2   3   4   5

b =

   1   2   3   4   5   6   7   8   9
```

ملاحظة:

هنا تم إنشاء مصفوفتين، ولكن تذكر بأنك تستطيع دمج التعبيرين ضمن سطر واحد إذا لم تفصل بفواصل:

```
>> c = [b   a]

c =

   1   2   3   4   5   6   7   8   9   1   2   3   4   5
```

وبذلك تم إنشاء مصفوفة c مؤلفة من عناصر b متبوعة بعناصر a.

تكييف المصفوفة

بالاعتماد على المثال السابق، فان فصل العناصر بفراغات أو بفواصل عادية يحدد عناصر في أعمدة مختلفة، في حين أن استخدام الفاصلة المنقوطة يجعل العناصر واقعة في أسطر مختلفة.

مثال:

```
>> c = [1   2   3   4   5]

c =

   1   2   3   4   5
```

مصفوفة أفقية

مثال:

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

c =

   1
   2
   3
   4
   5
```

مصفوفة عمودية (كل عنصر في سطر)

مثال:

```
>> a = 1: 5

a =

   1   2   3   4   5
```

مثال:

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

لقد استخدمنا هنا إشارة المنقول (المدور) لتحويل السطر a إلى العمود b.

```
b =

   1

   2

   3

   4

   5
```

مثال:

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

تنزيل سطر آخر

```
g =

   1   2   3   4

   5   6   7   8
```

مصفوفة متكونة من سطرين وأربعة أعمدة

مثال:

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

كذلك فأن ضغط مفتاح Enter أو Return يخبرنا برنامج MATLAB بأن ينتقل إلى سطر جديد أثناء إدخال قيم المصفوفة.

مثال:

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

Error عدد الأعمدة غير متساوية

مثال:

```
>> c = [1: 5 ; 2: 2: 10 ;  7: -1: 3]

c =

   1  2  3  4  5

   2  4  6  8  10

   7  6  5  4   3
```

ملاحظة:

تقدم لغة MATLAB طريقة أخرى للإشارة إلى عناصر المصفوفة باستخدام رقم واحد فقط، ولفهم هذه الطريقة يجب التخيل بأن جميع عناصر المصفوفة مرتبة بشكل عمود واحد مكون من أعمدة المصفوفة من الأعلى إلى الأسفل (أي عناصر العمود الأول ثم الثاني ثم الثالث وهكذا).

```
>> c (8)

ans =

   6

>> c (3)

ans =

   7
```

العمليات الحسابية بين المصفوفة والعدد المفرد

تجري العديد من العمليات الحسابية كعملية الإضافة والطرح والضرب والقسمة بين العدد المفرد وبين جميع عناصر المصفوفة.

مثال:

```
>> g – 2          % المصفوفة g المعرفة سابقاً

ans =

    -1   0   1   2

     3   4   5   6

     7   8   9  10
```

وهنا طُرح من كل عنصر من عناصر المصفوفة g العدد 2.

مثال:

```
>> 2 * g – 1

ans =

     1   3   5   7

     9  11  13  15

    17  19  21  23
```

أما هنا فضرب كل عنصر من عناصر المصفوفة g بالعدد 2، ثم طُرح من كل عنصر من العناصر الناتجة الرقم 1.

مثال:

```
>> 2 * g / 5 + 1

ans =

    1.4   1.8   2.2   2.6
```

```
    3    3.4   3.8   4.2

    4.6   5    5.4   5.8
```

أما في هذه الحالة، فقد ضُرب كل عنصر من عناصر المصفوفة g بالعدد 2، ثم قُسّم الناتج على العدد 5 وبعدها أُضيف لها الواحد.

العمليات الحسابية بين المصفوفات

برنامج MATLAB يستخدم لإجراء الحسابات التقنية المتقدمة ويتميز MATLAB بكونه برنامجاً متخصصاً ، فهو يتعامل مع المعادلات الرياضية ،والتكاملات ،والتفاضلات ، والمصفوفات المختلفة بسرعة وسهولة ، ويعامل الأعداد المركبة بنفس الطريقة التي يعامل بها الأعداد العادية . ويمكن MATLAB المستخدم من رسم المعادلات الرياضية في الإحداثيات المختلفة ، ويضم المئات من الدوال الجاهزة التي توفر للمبرمج وقتاً وجهداً عند إنشاء البرامج . وتعد العمليات الحسابية على المصفوفات متساوية الأبعاد من جمع وطرح وضرب وقسمة من العمليات الأساسية في لغة MATLABواليك الأمثلة التالية:

```
>> g          % إعادة استخدام المصفوفة السابقة

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

>> 2 * (g – h)

ans =

     0     2     4    6

     6     8    10   12

    12  14   16   18
```

لاحظ أيضاً بأن العمليات الحسابية بين المصفوفات تعتمد نفس تسلسل أسبقية العمليات المعتمد عند إجراء العمليات الحسابية على الأعداد المفردة، ويمكن أيضاً استخدام الأقواس لكسر تلك الأولوية.

المصفوفات القياسية

يمكّنك برنامج MATLAB من إنشاء مصفوفات قياسية، وذلك لتمتع تلك المصفوفات بخواص وميزات خاصة، وتتضمن أيضاً المصفوفات التي جميع عناصرها صفرية أو مساوية للواحد، ومصفوفات الأعداد العشوائية والمصفوفات القطرية والمصفوفات التي عناصرها أعداد ثابتة.

```
>> ones(4)          (مصفوفة واحديه)
```

مصفوفة واحدية : وهي المصفوفة التي تتكون جميع عناصرها من الرقم واحد .

```
ans =

   1   1   1   1

   1   1   1   1

   1   1   1   1

   1   1   1   1
```

```
>> ones(2)

ans =

   1   1

   1   1
```

```
>> zeros(3,6)          (مصفوفة صفرية)
```

الأعمدة الأسطر (الصف)

المصفوفة الصفرية : وهي المصفوفة التي تكون كل عناصرها عبارة عن اصفار وتعتبر هذه المصفوفة هي المحايد الجمعي للمصفوفات .

```
ans =
    0    0    0    0    0    0
    0    0    0    0    0    0
    0    0    0    0    0    0

>> zeros(2,4)
ans =
    0    0    0    0
    0    0    0    0
```

مثال :

```
>> 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
>> size (g)                    (تحديد أبعاد مصفوفة)
ans =
        3   4
>> ones (size (g))
ans =
      1    1    1    1
```

1 1 1 1

1 1 1 1

ملاحظة: عندما يتبع اسم المصفوفة القياسية برقم مفرد مثل ones (n) أو zeros (n) فأن برنامج MATLAB ينشيء مصفوفات مربعة $n \times n$ تحتوي على أصفارا أو واحديه على الترتيب.

>> eye(3) (مصفوفة الوحدة)

مصفوفة الوحدة : وهي مصفوفة مربعة تكون جميع عناصر القطر الرئيسي لها الواحد الصحيح وباقي عناصرها الاخرى اصفار .

ans =

1 0 0
0 1 0
0 0 1

>> eye (4)

ans =

1 0 0 0

0 1 0 0

0 0 1 0

0 0 0 1

س // انشأ مصفوفة ٢*٣ باستخدام الايعاز eye ؟

ج//

>> I = eye (2,3)

I =

1 0 0

0 1 0

مثال :

```
Sz = [3,1]
I = eye (sz)
I =
1
0
0
>> rand (3)          (مصفوفة عشوائية)
ans =
    0.9501    0.4860    0.4565
    0.2311    0.8913    0.0185
    0.6068    0.7621    0.8214
>> rand (1, 5)
ans =
    0.4447    0.6154    0.7919    0.9218    0.7382
```

مثال:

```
>> d = pi;
>> d * ones (3, 4)
ans =
    3.1416    3.1416    3.1416    3.1416
    3.1416    3.1416    3.1416    3.1416
```

```
    3.1416    3.1416    3.1416    3.1416
```

```
>> d + zeros (3, 4)

ans =

    3.1416    3.1416    3.1416    3.1416

    3.1416    3.1416    3.1416    3.1416

    3.1416    3.1416    3.1416    3.1416
```

```
>> repmat (d, 3, 4)        (تكرار القيمة d بالأبعاد 3×4)

ans =

    3.1416    3.1416    3.1416    3.1416

    3.1416    3.1416    3.1416    3.1416

    3.1416    3.1416    3.1416    3.1416
```

ملاحظة: يمكن ان تكون d مصفوفة فتكون حينئذ تكرر مصفوفات وليس قيم.

التعامل مع المصفوفة

يمتلك برنامج MATLAB العديد من الطرق للتعامل مع المصفوفات، وهذه الخاصية هي أهم مميزات البرنامج، فما إن تُحدّد المصفوفة حتى يزودك البرنامج بأقوى طرق الإدخال، أو إعادة ترتيب بعض أجزاء المصفوفة عبر استعمال تعابير أو تعليمات محددة وممتعة، وتعتبر معرفة هذه التعليمات مفتاح الاستعمال الفعال لبرنامج MATLAB. ولشرح التعامل مع المصفوفات نأخذ الأمثلة التالية:

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

A =

    1   2   3

    4   5   6

    7   8   9

>> A (3, 3) = 0

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
```

جعل العنصر في الموقع (3, 3) صفراً.

جعل العنصر في الموقع (2, 6) تكون 1 وبما ان المصفوفة A لا تمتلك ستة أعمدة، لذلك سيقوم البرنامج بتوسيعها حسب الضرورة ويضع بقية العناصر صفراً وتكون مستطيلة.

```
>> A (2, 5) = 3

A =

    1    2    3    0    0

    4    5    6    0    3

    7    8    9    0    0

>> A (:, 4) = 4

A =

    1   2   3   4   0   0

    4   5   6   4   0   1

    7   8   0   4   0   0

>> A (:, 4) = [4; 4; 4]

A =

    1   2   3   4   0   0

    4   5   6   4   0   1

    7   8   0   4   0   0

>> A (:, 4) = [4  4  4]

     Error
```

جعل جميع عناصر العمود الرابع تكون 4

جعل جميع عناصر العمود الرابع تكون 4

بسبب عدم وجود فارزة منقوطة

<u>مثال:</u>

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

B =
```

```
    7   8   9

    4   5   6

    1   2   3

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

A =

    1   2   3

    4   5   6

    7   8   9

>> C = [A   B(:, [1  3])]

C =

    1   2   3   7   9

    4   5   6   4   6

    7   8   9   1   3
```

حصلنا على المصفوفة C عبر توسيع المصفوفة A بإضافة العمودين الأول والثالث من المصفوفة B.

مثال: تشكيل المصفوفة B بجعل المصفوفة A كمصفوفة عمود واخذ أعمدتها عمود بعد عمود.

```
>> B = A (:)

B =

    1
    4
    7
    2
    5
    8
    3
```

6
9

مثال:

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

>> B = reshape (A, 1, 9)

B =

   1   4   7   2   5   8   3   6   9
```

Reshape: ويستخدم هذا الامر لإعادة تشكيل مصفوفة بأبعاد جديدة، اي بنفس معلومات عدد الصفوف والاعمدة.

```
>> B = reshape (A, [1  9]);

B =

   1   4   7   2   5   8   3   6   9
```

في المثال أعلاه إيعاز تحويل أبعاد المصفوفة الثنائية 3*3 إلى مصفوفة أحادية 9*1

مثال:

```
>> A = B

A =

   1   2   3

   4   5   6

   7   8   9
```

- لديك المصفوفة التالية :

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

A =
```

1 4 7 10

2 5 8 11

3 6 9 12

س // كيف يتم جعل المصفوفة A تحتوي على صفين وستة اعمدة باستخدام الامر Reshape؟

ج //

الطريقة الاولى B = reshape(A,2,6)

B =

1 3 5 7 9 11

2 4 6 8 10 12

الطريقة الثانية B = reshape(A,2,[])

B =

1 3 5 7 9 11

2 4 6 8 10 12

س // كيف يتم جعل المصفوفة A تتكون من عمودين وستة صفوف باستخدام الامر Reshape؟

<u>حذف الأسطر و الأعمدة في المصفوفة :</u>يمكن حذف الأسطر و الأعمدة للمصفوفة باستعمال زوج من الأقواس المربعة بالشكل التالي : لحذف العمود الثاني من مصفوفة X نكتب ما يلي X(:,2)=[]،حيث إشارة (:) تعني جميع العناصر أي أن الدليل (2,:) يعني جميع العناصر في العمود الثاني:

>> X= [2 5 8;7 8 9;4 3 5;1 4 8]

X =

2 5 8

7 8 9

```
     4     3     5

     1     4     8
```

هنا تم حذف الصف الاول من المصفوفة

```
>>  X (1,:)=[]

X =

     7     8     9

     4     3     5

     1     4     8
```

تم حذف عناصر السطر الأول من المصفوفة X في المثال السابق و لكن لم يسبب ذلك أي خطأ بينما حذف عنصر واحد من المصفوفة يسبب ظهور خطأ، مثلاً إذا أردنا حذف العنصر الأول من العمود الثاني X(1,2)=[] سيسبب ذلك ظهور إشارة خطأ ...

```
» X(1,2)=[]

???  Indexed empty matrix assignment is not allowed.
```

س// كيف يتم حذف العمود الثاني من المصفوفة x

ج //

```
>> X(:,2)=[ ]

X =

     2     8

     7     9

     4     5

     1     8
```

تمت إعادة صياغة المصفوفة X عبر حذف كل اسطر العمود الثاني من المصفوفة X الأصلية، وعندما تضع أي عنصر مساوياً للمصفوفة الفارغة [ ]، فهذا يعني انك تريد حذفها من المصفوفة وتقليصها لتحافظ على العناصر المتبقية بعد الحذف.

مثال: إيجاد منقول (مدور) المصفوفة وإعادة تشكيلها بالتعليمة reshape.

```
>> B=[3 5;7 9;4 8]

B =

     3     5

     7     9
```

```
    4    8
>> C = B'
C =
    3    7    4
    5    9    8
>> reshape (B, 2, 3)
ans =
    3    4    9
    7    5    8
```

ملاحظة: تعمل تعليمة reshape عمود بعد عمود وللحصول على سطر بعد سطر نعمل المدور (transport).

مثال: هنا حذفنا السطر الثاني في المصفوفة C.

```
 >> C (2, :) = [ ]
C =
    3    7    4
```

مثال: استبدلنا عناصر السطر الثاني من المصفوفة A بعناصر C.

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

```
   7    8    9
```

س // استبدل عناصر الصف (السطر) الثالث للمصفوفة A بالقيم التالية (1 41 20)

ج //

```
>>A (3,:) = [ 20  41  1]

A =

   1    2    3

   3    7    4

  20   41    1
```

س // استبدل العمود الاول للمصفوفة A بالقيم التالية (22 65 90)

مثال:

```
>> x = - 4 : 4

x =

  -4   -3   -2   -1    0    1    2    3    4
```

هناك المصفوفات المنطقية الناتجة عن العمليات المنطقية. كما يمكن أيضاً استخدام المصفوفات المنطقية إذا كان حجمها مساويا لحجم المصفوفات المعنونة، ويتم في هذه الحالة الإبقاء على العناصر ذات القيمة (1) أي true وهي العناصر المحققة للشرط بينما يتجاهل العناصر (0) أي false وهي العناصر غير المحققة الشرط. ولنأخذ المثال التالي:

```
>> abs (x) > 1

ans =

   1    1    1    0    0    0    1    1    1

>> abs (x) <2
```

مثال:

```
ans =

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

القيمة المطلقة abs (Absolute value) : وتمثل القيمة المطلقة للأعداد الحقيقية .

```
>> y = x (abs (x) > 1)
```

هنا تم إنشاء المصفوفة y من تلك العناصر من المصفوفة x التي قيمتها اكبر من الواحد.

```
y =

   -4   -3   -2    2    3    4
```

ويمكن العمل مع المصفوفات الثنائية المنطقية كما عملنا مع الأحادية المنطقية، كما في المثال التالي:

```
>> K = [6   -4; 3   -5]

K =

    6   -4

    3   -5

>> x = abs (K) > 2

x =

    1    1

    1    1


>> x = abs (K) > 4

x =
```

```
    1     0

    0     1

>> y = K (x)

y =

     6

    -5

>> T = abs (K) > 3

T =

    1     1

    0     1
```

س // كيف يتم استخراج القيم للمصفوفة T ؟

ترتيب المصفوفة

عندما تعطى متجه من البيانات فان أهم عملية يمكن إن نحتاجها وتود تطبيقها هي الترتيب، ويؤمن الايعاز sort عملية الترتيب في لغة MATLAB، كما هو واضح في المثال التالي:

```
>> x = randperm(8)                    (إيعاز ترتيب الأرقام بصورة عشوائية)

x =

    7   5   2   1   3   6   4   8

>> y = sort (x)

y =

    1   2   3   4   5   6   7   8
```

وعندما تكون المصفوفة ثنائية البعد فان عملية الترتيب تتم بشكل مختلف وكما يلي:

(عمود بعد عمود)

```
  >> A = [randperm (6); randperm (6); randperm (6); randperm (6)]

A =

    1   2   5   6   4   3
    4   2   6   5   3   3
    2   3   6   1   4   5
    3   5   1   2   4   6

>> h = sort (A)

hs =

    1   2   1   1   3   3
    2   2   5   2   4   3
```

```
    3   3   6   5   4   5
    4   5   6   6   4   6
```

البحث عن مصفوفة جزئية :

من المفيد في بعض الأحيان إن تعرف موقع أو دليل العناصر التي تحقق شرطا معينا، والموجودة ضمن مصفوفة معينة. يقوم برنامج MATLAB بتحقيق هذه الغاية عبر الايعاز find، والذي يعيد لك دليل أو موقع العنصر الذي تكون نتيجة تحقيقه لشرط ما true، واليك المثال التالي:

```
>> x = -3: 3

x =

    -3   -2   -1   0   1   2   3

>> k = find ((x) > 1)

k =                          (الموقع)

   6    7

>> k = find (abs (x) > 1)

  k =

    1   2   6   7

>> y = x (k)

y =

    -3   -2   2   3

>> y = abs (x (k))

y =
```

3 2 2 3

مثال :

```
S = [11  0  33  0  55];
find(S)
ans =
    1
    3
    5
```

س // اوجد موقع قيم الاصفار في المصفوفة S ؟

ج//

```
 find(S == 0)
 ans =
    2
    4
```

مثال :

```
>> A = [0 0 3;0 0 3;0 0 3]
A =
    0    0    3
    0    0    3
    0    0    3
>> find((A)>2)
ans =
    7
    8
    9
```

الدالتين Max و Min :

يوفر برنامج MATLAB الدالتين max ، min الذين يوجدان اكبر واصغر عنصر في المصفوفة ومواقعهما.

في حالة المصفوفة الأحادية:

```
>> v = rand (1, 6)

v =

   0.9340   0.6787   0.7577   0.7431   0.3922   0.6555

>> max (v)

ans =

      0.9340

>> min (v)

ans =

      0.3922
```

في حالة كون المصفوفة ثنائية البعد:

```
>> W= [ 1 7 3 ; 4 2 9]

W =

   1   7   3

   4   2   9
```

>> R = max (W) يأخذ البرنامج اكبر العناصر قيمة من بين كل عمود، ويضع الارقام التي حصل عليها بالتتابع في متجه .

```
R =
```

4 7 9

مثال :

```
>> A = rand (4, 6)

A =

    0.1509   0.8537   0.8216   0.3420   0.7271   0.3704

    0.6979   0.5936   0.6449   0.2897   0.3093   0.7027

    0.3784   0.4966   0.8180   0.3412   0.8385   0.5466

    0.8600   0.8998   0.6602   0.5341   0.5681   0.4449

>> x=max (A)

x =

    0.8600   0.8998   0.8216   0.5341   0.8385   0.7027
```

ملاحظة:

```
>> max (A');          (اكبر عنصر لكل سطر)

>> [m] = min (A)

m =

    0.1509   0.4966   0.6449   0.2897   0.3093   0.3704
```

ملاحظة:

```
>> min (A');          (اصغر عنصر لكل سطر)
```

ملاحظة: اكبر عنصر في مصفوفة ثنائية البعد.

```
>> g = max (max( A))
```

ملاحظة: نفس الشيء لحساب المجموع sum.

```
>> z = sum (sum (A));
```

حجم المصفوفة

إذا أردت أن تعرف حجم أو بعد مصفوفة أحادية أو ثنائية أو ثلاثية البعد غير معروفين وكنت بحاجة لحجمها لإجراء بعض العمليات الرياضية، فان برنامج MATLAB يمكنك من خلال الايعاز length و size و numel وغيرها واليك الأمثلة التالية:

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

A =

    1   2   3   4

    5   6   7   8
```

الدالة Size : تقوم هذه الدالة بعرض أبعاد المصفوفة كما في المثال :

```
>> S = size (A)

S =

    2   4
```

يعبر العنصر الأول عن عدد الأسطر (الصفوف) (2) بينما يعطي العنصر الثاني عدد الأعمدة (4).

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

r =                    تمثل عدد الصفوف

    2

c =                    تمثل عدد الاعمدة

    4
```

يعيد الايعاز numel العدد الكلي لعناصر مصفوفة فمثلاً:

```
>> numel (A)

ans =

      8
```

بينما يعيد الايعاز length عدد العناصر الموجودة ضمن البعد الأطول للمصفوفة، كما يلي:

```
>> length (A)
ans =
      4
>> B = -3: 3
B =
    -3   -2   -1   0   1   2   3
>> length (B)
ans =
      7
```

المصفوفات متعددة الإبعاد

لقد شرحنا في الفصل السابق المصفوفات أحادية وثنائية الأبعاد والعمليات التي تجري عليها. يدعم برنامج MATLAB المصفوفات متعددة الأبعاد (أي n-D arrays) وذلك نفس الايعازات وتقنيات العنونة المطبقة على المصفوفات أحادية وثنائية البعد. وبشكل عام، يرقم البعد الثالث عبر صفحات (pages)، ولذلك تمتلك المصفوفات ثلاثية البعد اسطرا وأعمدة وصفحات، حيث تتألف كل صفحة من مصفوفة ثنائية البعد ذات اسطر وأعمدة، ويجب أن تمتلك كل صفحة عددا متساويا من الأسطر والأعمدة والعكس بالعكس في كل صفحة.

ليس هناك حد لعدد الأبعاد في المصفوفات، ولكننا سنستخدم مصفوفات ثلاثية الأبعاد في هذا الفصل بسبب سهولة تخيلها وإظهارها.

تركيب المصفوفة

يمكن إنشاء المصفوفة المتعددة الأبعاد بطرق مختلفة، واليك بعضها:

```
>> A = zeros (4, 3, 2)
A (:, :, 1) =
        0   0   0
        0   0   0
        0   0   0
        0   0   0
A (:, :, 2) =
        0   0   0
        0   0   0
        0   0   0
        0   0   0
```

تتألف هذه المصفوفة الصفرية من أربعة اسطر وثلاثة أعمدة وصفحتين، ولقد ظهرت الصفحة الأولى ثم الصفحة الثانية.

مثال:

```
>> B (:, :, 1) = zeros (2, 3);
>> B (:, :, 2) = ones (2, 3);
>> B (:, :, 3) = 4;
>> B
B (:, :, 1) =
        0   0   0
        0   0   0
B (:, :, 2) =
        1   1   1
```

```
     1   1   1
B (:, :, 3) =
     4   4   4
     4   4   4
```

حجم المصفوفة

الايعاز size يعيد بعد المصفوفة وفق كل أبعادها كما شرحنا سابقاً.

```
>> [r, c, p] = size (C)
r =
    2
c =
    3
p =
    3
>> numel (C)                إيجاد عدد عناصر المصفوفة
ans =
      18
>> length (size (C))        إيجاد طول اكبر بعد بالمصفوفة
ans =
      3
```

مصفوفة الخلايا Cell Arrays

تعتبر مصفوفات الخلايا مصفوفات في لغة MATLAB تكون عناصرها عبارة عن خلايا، وتتضمن كل خلية نوعاً من البيانات قد تكون مصفوفات عددية أو رمزية أو كائنات بسيطة أو مصفوفات خلايا أخرى. فمثلاً قد تحوي خلية من مصفوفة الخلية مصفوفة عددية وتحوي الخلية الأخرى مصفوفة رمزية، بينما تحوي الثالثة على أعداد عقدية (يسمح باستخدام مصفوفات بأنواع بيانية مختلفة (غير متجانسة)) كما ويمكن إنشاء مصفوفات الخلايا بأي بعد كان كما هي الحال مع المصفوفات العددية، ولكن معظم مصفوفات الخلايا تكون عبارة عن مصفوفات أحادية البعد.

تنشأ مصفوفات الخلايا عبر استخدام تعابير الإسناد أو عبر إعادة تقسيم المصفوفة بالإيعاز cell، ثم نقوم بإسناد البيانات إلى الخلايا.

هناك طريقتان مختلفتان للوصول إلى الخلايا. وإذا أردت استخدام رموز مصفوفة قياسية للدلالة على المصفوفة، يجب عليك أن تحيط الخلية بأقواس مجموعة { }، إذ إن برنامج MATLAB يستخدم هذه الأقواس لتعريف مصفوفات الخلايا، واليك الأمثلة التالية:

```
>> cell (1, 1) = {[1   2   3; 4   5   6; 7   8   9]};

>> cell (1, 2) = {2 + 3i};

>> cell (2, 1) = {'Ali Ahmed'};

>> cell (2, 2) = {12: -2: 0};

>> cell

cell =

    [3×3 double]       [2.0000 + 3.0000i]

    'Ali Ahmed'        [1×7 double]
```

لاحظ إن برنامج MATLAB يظهر المصفوفة كمصفوفة خلية بعداها 2×2 ولكن ذلك لا يظهر محتويات الخلية، وإنما يظهر البرنامج محتويات الخلية بشكل أساسي إذا لم تأخذ هذه

المحتويات حجماً كبيراً، كما ويوصف محتويات الخلية إذا أخذت حجماً معقولاً. إن وجود أقواس مجموعات على الجانب الأيمن من المساواة يدل على إن المشار إليه هو خلية وليس قيما عددية وهذا ما يسمى *بفهرسة الخلية (cell indexing)*، وسينشئ التعابير التالية مصفوفة الخلية نفسها.

مثال:

```
>> x = rand (3, 3);
>> y = rand (3, 3);
>> z = rand (3, 3);
>> w {1} = x;
>> w {2} = y;
>> w {3} = z;
>> w
ans =
    [3×3 double]    [3×3 double]    [3×3 double]
```

مثال:

```
>> x {1} = rand (3, 3);
>> x {2} = rand (3, 3);
>> x {3} = rand (3, 3);
      .
      .
      .
      .
      .
>> x {9} = rand (3, 3);
>> x {1}
```

```
ans =

    0.8462    0.6721    0.6813
    0.5252    0.8381    0.3795
    0.2026    0.0196    0.8318
```

العنصر الموجود في السطر الثاني والعمود الثاني في مصفوفة (الخلية) الأولى

```
>> x {1} (2, 2)

ans =

    0.8381
```

يجبر الإيعاز celldisp برنامج MATLAB على إظهار محتوى الخلايا بالنموذج العادي، واليك المثال التالي الذي يوضح ذلك:

```
>> celldisp (A)
A (1, 1) =
        1    2    3
        4    5    6
        7    8    9
  A (2, 1) =
        Ali Ahmed
  A (1, 2) =
        2.0000 + 3.0000i
  A (2, 2) =
        12    10    8    6    4    2    0
```

جمل الإدخال والإخراج

جمل الإدخال

هناك عدة صيغ للإدخال بالإضافة إلى عملية التنسيب اهمها:

تعليمة input:

مثال (١):

```
>> x = input ('enter x: ')

enter x:
```

مثال (٢): إدخال أسماء رمزية.

```
z = input ('enter name', 's');
```

للدلالة على إدخال string

جمل الإخراج

هناك عدة صيغ للإخراج منها:

١- تعليمة disp:

مثال (١):

```
>> d = 15;

>> disp (d);

15
```

مثال (٢):

```
>> a = 'ali';

>> disp (a);

ali
```

مثال (٣):

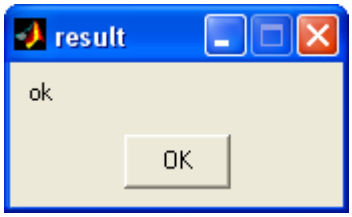
```
>> disp ('computer');

Computer
```

٢- تعليمة msgbox:

```
>> msgbox ('ok', 'result')
```

عنوان الصندوق

الشيء المطلوب طباعته (نوع بياني رمزي)



الجمل الشرطية

يدعم برنامج MATLAB العمليات المنطقية والمقارنة مثلما يدعم العمليات الرياضية، وتهدف العمليات والمعاملات المنطقية الحصول على أجوبة للأسئلة التي يجاب عنها بصح أو خطأ(True/False) .

تعتبر لغة MATLAB في تعاملها مع جميع التعابير المنطقية وعمليات المقارنة إن أي عدد غير صفري هو True ويعتبر الصفر False، كما ويكون إخراج جميع التعابير المنطقية

وعمليات المقارنة عبارة عن مصفوفات منطقية تحوي العدد واحد من اجل True والعدد صفر من اجل False.

وتعتبر المصفوفات المنطقية نوعاً خاصاً من المصفوفات العددية، كما يمكن عنونة المصفوفة المنطقية بنفس طريقة عنونة باقي المصفوفات التي استخدمها سابقاً ضمن التعابير العددية.

معاملات المقارنة (العوامل العلائقية) : Relational Operators

تتضمن معاملات المقارنة كل الإشارات المقارنة الشائعة والمدرجة في الجدول التالي:

معامل المقارنة	الوصف
<	أصغر من
<=	أصغر أو يساوي
>	أكبر من
>=	أكبر أو يساوي
==	إشارة المساواة (لكي نميزها عن =)
~=	اشارة عدم المساواة

يمكن استخدام معاملات المقارنة للمقارنة بين مصفوفتين لها نفس الحجم، أو للمقارنة بين مصفوفة وعدد مفرد وتتم هذه الحالة مقارنة كل عنصر من المصفوفة مع العدد المفرد، وتكون المصفوفة الناتجة بنفس حجم المصفوفة التي تمت مقارنتها كما يبينه المثال التالي:

مثال (١):

```
>> a = 1;        b = 5;
>> x = a > b
x =
    0
```

```
>> 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
>> k = A > 4
k =
    0   0   0   0  1   1   1   1   1
```

لقد أوجدنا العناصر من A التي هي أكبر من 4، وتظهر الاصفار في المصفوفة الناتجة في مواقع العناصر عندما $A \leq 4$، بينما يظهر الرقم 1 عندما $A > 4$.

```
>>m  = (A == B)
m =
    0   0   0   0   0   0   0   0   0
```

لقد تم هنا إيجاد عناصر A التي تساوي العناصر في المصفوفة B.

ملاحظة:

لاحظ بان الإشارتين (=) و (==) تعنيان شيئاً مختلفاً، حيث يقوم (==) بمقارنة متغيرين وتعيد العدد واحد إذا كانا متساويين وصفراً إذا لم يكونا متساويين، بينما تستخدم (=) لإسناد إخراج العملية إلى متغير.

مثال (١): لتوليد مصفوفة أحادية منطقية عناصرها واحدات (في حالة اكبر من thr) واصفاراً (في حالة اصغر من أو تساوي thr).

```
>> inddent = [10   17   22   0   7   3   2];
>> thr = 7;
>> y = (inddent > thr)
y =
```

1 1 1 0 0 0 0

المعاملات المنطقية (العوامل المنطقية) :Logical Operators

توفر المعاملات المنطقية طريقة لدمج أو نفي تعابير المقارنة، ويظهر الجدول التالي المعاملات المنطقية الموجودة في لغة MATLAB:

المعامل المنطقي	الوصف
&	AND (و)
|	OR (أو)
~	NOT (نفي)

- بعض الأمثلة على استخدام المعاملات المنطقية:

```
>> a = 1;
>> b = 5;
>> x = a ~= b
x =
   1
>> b = (1 == 1) & (2 ~= 3)
b =
   1
>> b = (1==1) | (2 ~= 3)
b =
   1
>> A = 1: 9;            B = 9 - A;
>> tf = A > 4
tf =
```

0 0 0 0 1 1 1 1 1

حيث قام بإيجاد عناصر A التي قيمها اكبر من 4

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

لقد قام البرنامج بقلب النتيجة السابقة، وتعني استبدال مواقع الاصفار والواحدات.

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

حيث تعيد هذه العبارة العدد واحد عندما يكون العنصر من A اكبر من 2 واقل من 6.

أسبقية المعامل

يقوم برنامج MATLAB بإيجاد قيمة تعبير مستنداً إلى مجموعة من القواعد المنظمة لأسبقية المعامل، وتحسب المعاملات ذات الأسبقية العليا قبل المعاملات ذات الأسبقية الدنيا، وتقيم المعاملات ذات الأسبقية المتساوية من اليسار إلى اليمين. ويشرح الجدول التالي قواعد أسبقية المعامل التي يعتدها برامج MATLAB.

المعامل	مستوى الأسبقية
الأقواس ( )	الأعلى
المدور(')، القوة (^، .^)	
إشارة النفي (~)	
الضرب (*، .*)، القسمة (/، ./)	
الجمع (+)، والطرح (-)	

معامل النقطتين المتعامدتين (:)	
أصغر من (>)، واصغر أو يساوي(=>)، اكبر من (<)، اكبر من أو يساوي(=<)، المساواة (==)،عدم المساواة (=~)	
الجمع المنطقي(&) AND	
المعامل المنطقي (|) OR	الأدنى

الصيغة IF-ELSE-END

قد نحتاج إلى حساب مجموعة من أوامر استناداً إلى إخراج ناتج عن اختبار شرطي. وتنفذ هذه التعليمة في لغة MATLAB عبر استخدام الصيغة if-else-end وكما يلي:

```
if expression
   (commands)
end
```

وستنفذ الأوامر (commands) الواقعة بين العبارتين if وend إذا كانت قيمة التعبير (expression) تكون true. واليك المثال التالي:

```
>> x = 10;
>> if  x == 10
     disp ('ok')
   end;
```

وإذا كان لدنيا خياران، فتصبح الصيغة if-else-end كما يلي:

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

حيث ستنفذ المجموعة الأولى من الأوامر في حال امتلك التعبير expression القيمة true، بينما تنفذ المجموعة الثانية إذا امتلك التعبير expression القيمة false.

وإذا كانت هناك عدة حالات، فستأخذ التعبير if-else-end الشكل التالي:

```
if expression1
  (commands evaluated if expression1 is true)
elseif expression2
  (commands evaluated if expression2 is true)
elseif expression3
  (commands evaluated if expression3 is true)
elseif expression4
  (commands evaluated if expression4 is true)
.
.
.
else
```

(commands evaluated if no other expression is true)

end

واليك الأمثلة التالية:

مثال (1):

```
a=[1 2 3; 4 5 6; 6 7 8]
if a(2)==4
    disp ('ok')
else
    disp ('end')
end
```

مثال (2):

```
>> x = 10;
>> if  x == 10
  msgbox ('ok', 'result');
end;
```

مثال (3):

```
>> if  x ==  10
     msgbox ('ok', 'result');
   else
     msgbox ('no', 'result');
   end;
```

مثال (4):

```
>> x = 11;
>> if  x == 1
     disp ('1');
   elseif  x == 2
     disp ('2');
```

```
 else
    disp ('3');
end;
```

المخرجات

3

جمل الدوران والتكرار

توفر لغة MATLAB مجموعة من جمل الدوران والتكرار وهي:

جملة for

تقوم حلقات for بإعادة تنفيذ مجموعة من الأوامر لعدد معين من المرات وبخطوة معينة، وتعطى الصيغة العامة لحلقة for كما يلي:

```
for  i = x1: x3: x2

  (commands)

end;
```

حيث يعاد تنفيذ الأوامر (commands) الواقعة بين عبارتي for و end من القيمة الابتدائيةx1 إلى القيمة النهائية x2 وبزيادة مقدارها x3. كما في المثال التالي:

مثال:

```
>> for n = 1: 10

      Disp (n)

   end;
```

ويمكن تفسير الدوران أعلاه كما يلي:

من اجل كل قيمة لـ n من 1 إلى 10 يجب حساب قيمة العبارة الموجودة حتى عبارة end التالية، تكون قيمة n في الدورة الأولى n = 1، وتكون في الدورة الثانية n = 2 وهكذا حتى تصل إلى n = 10.

أمثلة:

```
>> for i = 1: 10
      disp (i);
   end;
```

الإخراج

```
1
2
3
.
.
10
```

```
>> for i = 0: 2: 10
     disp (i);
   end;
```

الإخراج

```
0
2
4
6
8
10
```

```
>> for i = 10: -2: 1
      disp (i);
    end;
```

الإخراج

10

8

6

4

2

```
>> for i =1: 10
    for j = 1: 10
        mult (i, j) = i * j;
    end;
end;
```

(طبع جدول الضرب)

1	2	3	4	5	6	7	8	9	10
2	4	6	8	10	12	14	16	18	20
3	6	9	12	15	18	21	24	27	30
4	8	12	16	20	24	28	32	3 6	40
.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.
10	20	30	40	50	60	70	80	90	100

جملة WHILE

تُجري حلقات while عمليات الحساب عدداً غير محدد من المرات على عكس حلقات for التي تؤدي عدداً معيناً من التمريرات، ويمكن كتابة الصيغة العامة لحلقة while كما يلي:

```
while expression
    (commands)
end;
```

ستنفذ مجموعة الأوامر(commands) الواقعة بين العبارتين while و end طالما أن كل العناصر ضمن expression تمتلك قيماً صحيحة (true)، وعادةً ما تكون نتيجة expression عدداً مفرداً.

مثال (1):

```
>> x = 1;
>> while x < 25
    disp (x);
   x = x + 1;
   end;
```

الإخراج

1

2

3

.

.

24

مثال :

اجمع الاعداد من 1 الى 10 وبزيادة مقدارها 0.5

- while

```
s=0;
i=1;
while i<=10
s=s+i;
i=i+0.5;
end
disp(s)
```

- For

```
- s=0;
- for i=1:0.5:10
-     s=s+i;
- end
- disp(s)
```

مثال : اكتب برنامج بلغة ماتلاب لحساب :

1- $1^2 + 2^2 + 3^2 + \ldots\ldots + 1000^2$

الحل :

```
s=0;
for k=1:1000
    s=s+k^2;
end
disp(s)
```

ملفات البيانات الخاصة ببرنامج MATLAB

يمكن تخزين المتغير الموجود في ساحة عمل برنامج MATLAB، وفق صيغة خاصة ببرنامج MATLAB، وذلك عن طريق استخدام الأمر save كما يلي:

```
>> save
```

وبذلك يتم خزن جميع المتغيرات الموجودة في ساحة العمل (Workspace) في ملف ذي صيغة ثنائية باسم matlab.mat يوضع في المجلد الحالي. وتحافظ هذه الملفات ذات صيغة الثنائية، والخاصة ببرنامج MATLAB، على كامل القيم وبدقة مضاعفة، كما وتخزن أسماء المتغيرات بنفس الدقة، ولا تعتبر ملفات MAT-files ذات أصول مستقلة، إنما هي متوافقة تماماً مع بقية أنواع الملفات الموجودة في برنامج MATLAB، إذ نستطيع تخزين أي متغير وفق نوع من الملفات وفتحة من قبل الأنواع الأخرى دون إجراء أية معالجة خاصة للملف.

ويمكن أن يستخدم الأمر save لتخزين متغيرات معينة كما في المثال التالي:

```
>> save   var1   var2   var3
```

أي قم بتخزين المتغيرات var1 وvar2 و var3 ضمن الملف matlab.mat، ويمكن أن نحدد اسم الملف كوسيط أول للأمر save كما يلي:

```
>> save   filename   var1   var2   var3
```

ضمن ملف اسمه var1, var2, var3 وتفسر التعليمة السابقة كما يلي: خزّن المتغيرات filename.mat.

ويعاكس الأمر load الأمر save إذ يفتح هذا الأمر ملفات البيانات التي تم إنشاؤها بالأمر save كما يلي:

```
>> load
```

وهي تعني حملّ كل المتغيرات التي تجدها ضمن الملف matlab.mat حيثما وجدته أولا سواءً في المجلد الحالي أو في مسار البحث لبرنامج MATLAB. ويتم تخزين أسماء المتغيرات المخزونة في الملف matlab.mat في ساحة العمل، وستحمّل فوق المتغيرات ذات الأسماء المطابقة لها في حال وجودها.

ولتحميل متغيرات معينة من ملف ذي لاحقة (MAT-file) يجب ان نذكر اسم الملف وقائمة المتغيرات كما يلي:

```
>> load   filename   var1, var2, var3
```

لقد تم هنا فتح الملف filename.mat وحملت المتغيرات var1, var2, var3 إلى ساحة العمل.

ايعازات المجموعات والبتات والايعازات القاعدية

ايعازات المجموعات

نستطيع تقييم المصفوفات على إنها مجموعات لأنها تجميع منتظم لعدد من القيم وانطلاقاً من هذا الفهم، يقدّم لك برنامج MATLAB عدة توابع لاختبار ومقارنة المجموعات، ويقدم لك المثال التالي ابسط اختبار للمساواة:

```
>> a = rand (2, 5);

>> b = rand (2, 5);

>> isequal (a, b)

ans =

     0

>> isequal (a, a)

ans =

     1
```

ويقدم لك المثال التالي الايعاز unique بحذف العناصر المتكررة من وسط الإدخال:

```
>> a = [2: 2: 8; 4: 2: 10]

a =

   2   4   6   8

   4   6   8   10

>> unique (a)

ans =

      2

      4

      6

      8

     10
```

ويمكن تحديد مجموعة العناصر المشتركة بين وسيطين عبر استخدام الايعاز ismember كما يلي:

```
>> a = 1: 9

a =

   1   2   3   4   5   6   7   8   9

>> b = 2: 2: 9

b =

   2   4   6   8
```

```
>> ismember (a, b)

ans =

     0   1   0   1   0   1   0   1   0

>> ismember (b, a)

ans =

     1   1   1   1
```

كذلك الايعاز union لاتحاد مجموعتين.

```
>> union (a, b)

ans =

     1   2   3   4   5   6   7   8   9
```

كذلك إيعاز intersect لتقاطع مجموعتين.

```
>> intersect (a, b)

ans =

     2   4   6   8
```

كذلك إيعاز setdiff للفضلة بين مجموعتين.

```
>> setdiff (a, b)

 ans =

      1   3   5   7   9
```

ملاحظة:

يمكن إجراء العمليات السابقة على مصفوفات رمزية أو مصفوفات خلايا.

عندما يتوجب علينا تنفيذ أوامر اعتماداً على استخدام متكرر لاختيار كمي لوسط ما، عندها من السهل استخدام الصيغة switch-case التي لها الصيغة العامة التالية:

```
switch expression
  case test-expression1
     (commands1)
  case test-expression2
     (commands2)
   otherwise
      (commands3)
end
```

يجب أن يكون expression هنا أما عدداً مفرداً أو سلسلة رمزية. يقارن التعبير expression الموجود في الصيغة السابقة بالتعبيرtest-expression1 الموجود في عبارة case الأولى. وإذا تساوى التعبيران، سيتم تنفيذ الأوامر (commands1) وتخطي التعليمات الواقعة بعدها حتى العبارة end. أما إذا لم يتحقق الشرط الأول، فسيختبر الشرط الثاني، حيث سيقارن expression في المثال السابق مع العبارات test-exoression2 الموجودة في عبارة case الثانية. وإذا تساوى التعبيران، سيتم تنفيذ (commands2) وتهمل بقية العبارات حتى عبارة end. إذا لم تحقق أي عبارة case المساواة مع التعبير expression، عندها ستنفذ الأوامر (commands3) التي تلي العبارة otherwise.

لاحظ من الشرح الذي أوردناه عن صيغة switch-case بأنه سيتم تنفيذ أحدى مجموعات الأوامر المكونة للصيغة switch-case واليك الأمثلة التالية:

مثال (١):

```
 x = 1;
switch x
   case {1, 2, 3, 4, 5}
      disp ('1..5');
   case {9, 10}
      disp ('9..10');
   otherwise
      disp ('this is impossible');
end;
```

المخرجات 1..5

مثال (٢):

```
x = 2.7;
units = 'm';
switch units
  case {'inch', 'in'}
    y = x * 2.54;
  case {'meter', 'm'}
    y = x / 100;
  case {'feed', 'ft'}
    y = x * 2.54 / 12;
  case {'millimeter', 'mm'}
    y = x * 10;
  case {'centimeter', 'cm'}
    y = x;
```

```
  otherwise
    disp (['Unknown Units:'    units]);
end;
```

الإخراج

y = 0.027