

عمليات المقارنة

توجد ستة عمليات منطقية تستخدم لغرض المقارنة بين المصفوفات , و من الممكن ان تكون المقارنة بين قيمة عددية و عناصر مصفوفة أو عناصر متجه أو قيمة عددية اخرى

الرمز	العلاقة
<	اصغر من
<=	اصغر من أو يساوي
>	اكبر من
>=	اكبر من أو يساوي
==	يساوي
~=	لا يساوي

```
>>a=[1 2 6;2 1 2;3 6 4];
```

```
>>b=[7 1 8; 1 7 4; 2 3 9];
```

```
>>a
```

```
A=
```

```
1 2 6
2 1 2
3 6 4
```

```
>>b
```

```
b =
```

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

```
>> h=a>b
```

```
h=
```

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

```
>> h=a<b
```

```
H=
```

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

```
>> h1=(a<=b)
```

```

h1=
1 0 1
0 1 1
0 0 1

>> h2=(a~b)
h2=
1 1 1
1 1 1
1 1 1
>> h3=(a==b)
h3=
0 0 0
0 0 0
0 0 0

>> h4= a > 3
h4=
0 0 1
0 0 0
1 1 0

>> h5= b < 4
h5=
0 1 0
1 0 0
1 1 0

```

كما ان العمليات المنطقية يمكن تطبيقها على المتجهات

```

>> k= [4 9 12];
>> p=[13 2 18];
>> h=k>p
h=
1 0 1
>>h= p<4
h=
0 1 0
>>h=p ==13
1 0 0

```

البواب المنطقية

توجد ثلاث عمليات رئيسية للمقارنة بين مصفوفتين أو متجهين أو مصفوفة مع قيمة عددية.

الرمز	العملية	الشرح
&	And	يوضح عمل بوابة AND المنطقية
	Or	يوضح عمل بوابة OR المنطقية
~	not	يوضح عمل بوابة NOT المنطقية

```
>> a=[1 0 1];
>> b=[0 1 1];
>> h1= a&b
h1=
0 0 1
>> h2= a|b
h2=
1 1 1
>> h3=~a
h3=
0 1 0
>> h4=~b
h4=
1 0 0
```

نفس العمليات يمكن تطبيقها على مصفوفات ذات بعدين

H.W

إذا كان $a=[1 0; 0 1]$ و $b=[0 0; 1 1]$. قم بتطبيق العمليات المنطقية الثلاثة بين المصفوفتين

M-file

هي وسيلة لإدخال الأوامر ولكن ليس من خلال نافذة الأوامر, ولكن ماذا قد يختلف في هذه الوسيلة الجديدة في إدخال الأوامر؟

1 - في عملية إدخال الأوامر التي كنا نستخدمها, إذا أردنا تعديل عنصر أو أكثر كان يجب

إعادة إدخال الأمر من جديد.

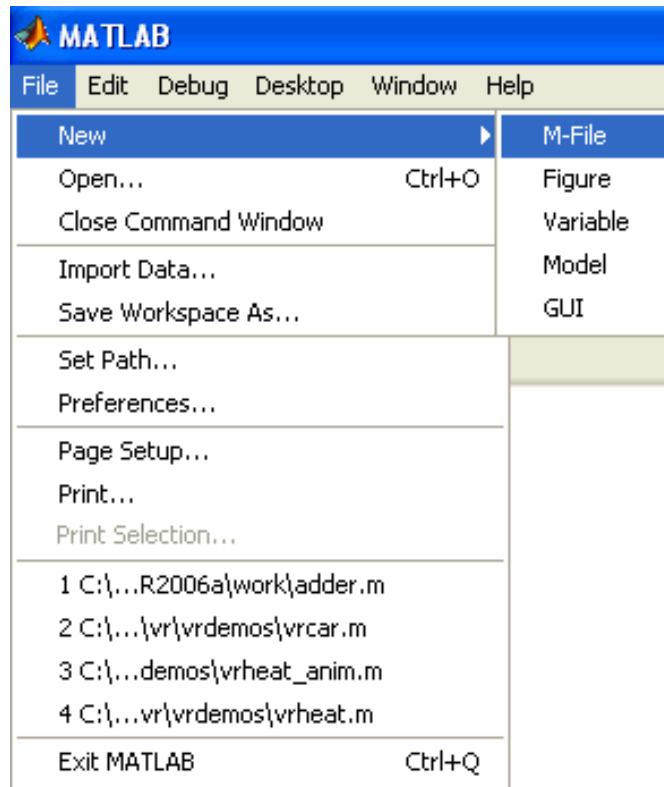
2- إذا وجد خطأ, فيجب كتابة الأمر من جديد

3 - إذا كتبنا برنامج كبير, وأردنا إعادة العملية مرة أخرى يجب إدخال جميع الأوامر من جديد وبنفس الترتيب.

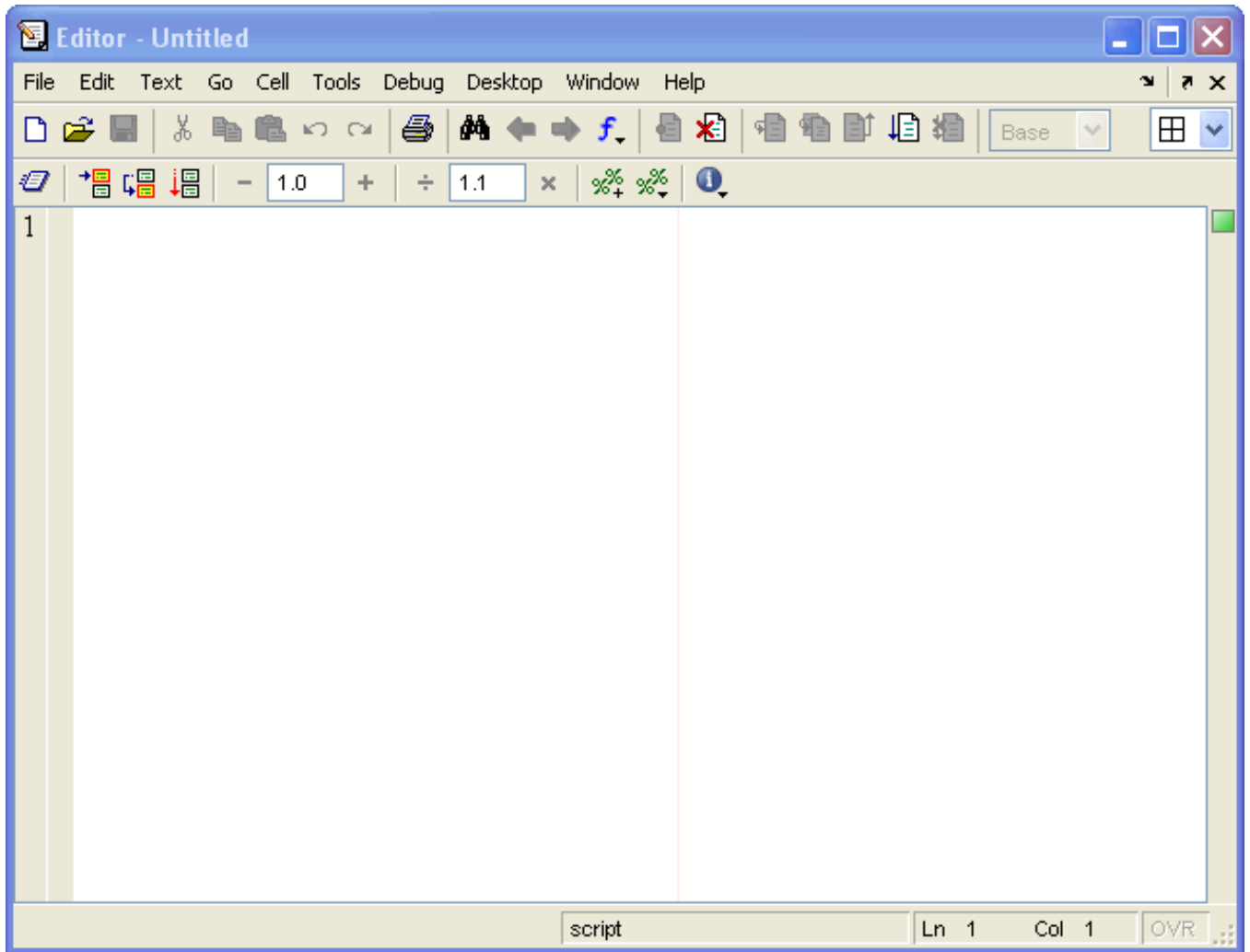
4 - إذا حدث خطأ في ترتيب الأوامر لهذا البرنامج الكبير ستقوم بإعادة الإدخال الأوامر من البداية مرة أخرى.

5 - يصعب عمل عملية تصحيح للأخطاء Debugging

وهذا بالطبع يستغرق وقتاً كبيراً هذا بالإضافة إلى الملل الذي يحدث للمستخدم وطبعاً حلاً لهذه المشكلة, تم عمل بما يسمى M-File والتي تعطي القدرة على كتابة البرنامج كاملاً أولاً بدون تشغيل, وبعد الانتهاء منه يتم تشغيله, هذه الخاصية تعطي القدرة على تعديل القيم دون الحاجة إلى كتابتها مرة أخرى, أو إعادة إدخال الأوامر التي تعتمد على هذا الأمر . فكيف يتم تشغيل تلك الخاصية؟ اتبع الصورة التالية



وبالتالي ستظهر نافذة جديدة, تأخذ الشكل التالي

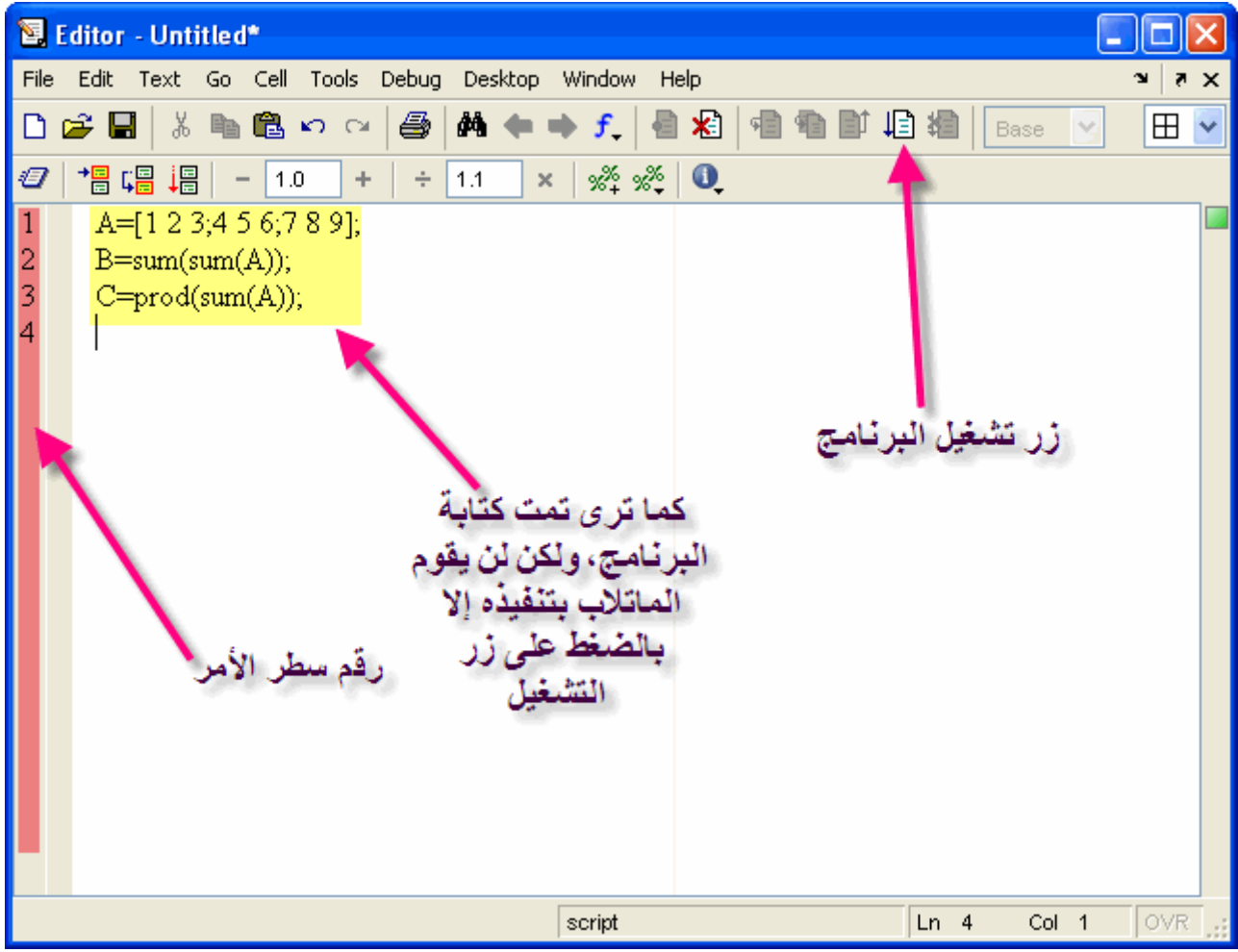


بالنسبة الى MATLAB R2013a

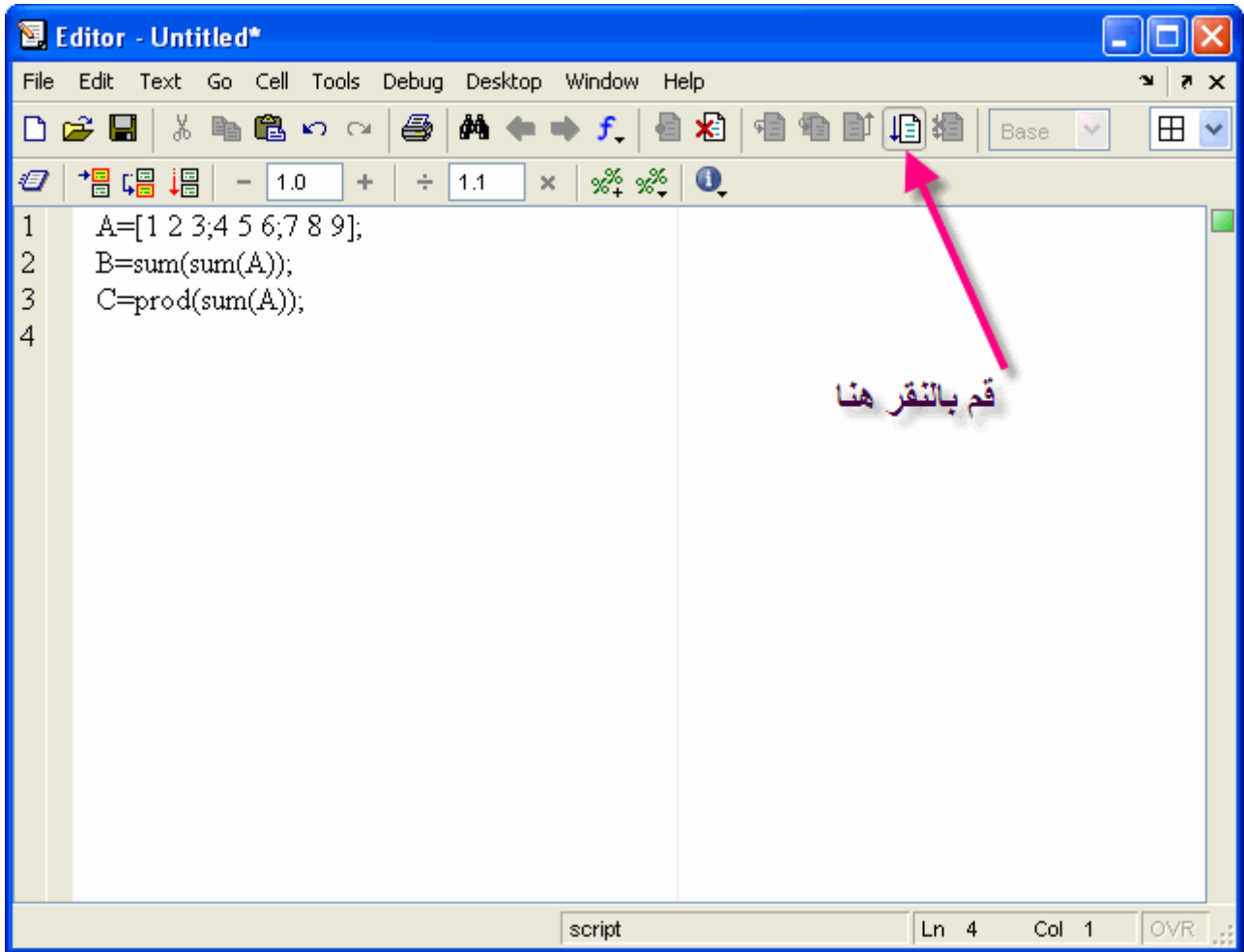
يمكن فتح نافذة MFile من خلال الضغط على الامر New Script في النافذة الرئيسية او من خلال Ctrl+N في لوحة المفاتيح.

نافذة M-File :

سنقوم الآن بالتعرف على نافذة M-File, أنظر الصورة التالية :

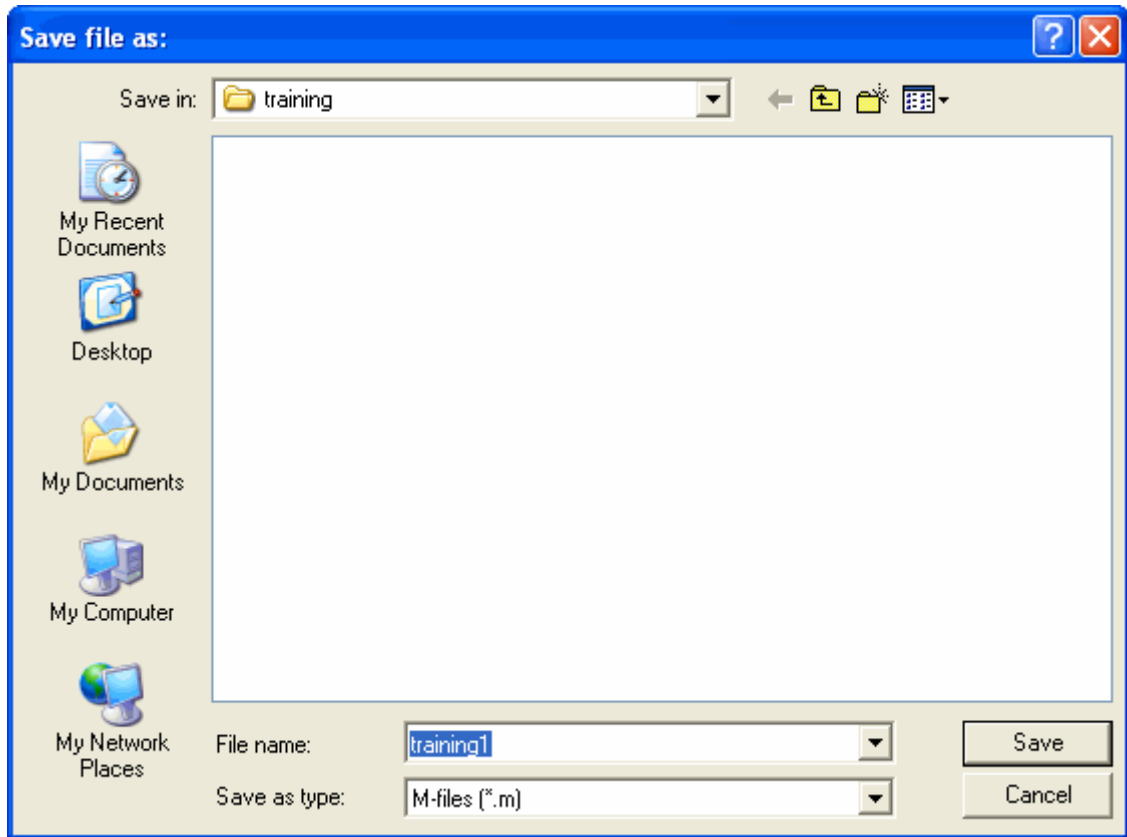


- ولكن عند الضغط على زر التشغيل, سيطلب الماتلاب بحفظ البرنامج, ولكن يشترط الآتي عند حفظ البرنامج
- 1 - أن لا يبدأ بأرقام
 - 2 - أن لا يكون أمراً معرفاً في الماتلاب
 - 3 - أن لا يحتوي الاسم على مسافات فاصلة
 - 4 - أن لا تحتوي على رموز خاصة مثل *, &, -, +
- يجب مراعاة تلك الشروط و إلا لن يقوم الماتلاب بتنفيذ البرنامج فالنقم بتنفيذ المثال المكتوب الآن في النافذة السابقة
- 1 - يتم الضغط على زر التشغيل كما هو واضح في الصورة التالية



او من خلال قائمة Debug نختار الامر Save and Run

2 - سيطالبنا الماتلاب بحفظ البرنامج أولاً, ولنسميه training1



3 - ستظهر القيم في كلاً من Command Window and Workspace
and Workspace

The image displays the MATLAB software interface. At the top, the title bar reads "MATLAB" and "EN Eng". The menu bar includes "File", "Edit", "View", "Graphics", "Debug", "Desktop", "Window", and "Help". The current directory is "C:\Documents and Settings\Eng.Ahmed Afifi\Desktop\training".

The **Workspace** window shows three variables:

Name	Value	Class
A	[1 2 3; 4 5 6; 7 8 9]	double
B	45	double
C	3240	double

The **Editor** window shows the following code:

```

1 - A=[1 2 3;4 5 6;7 8 9]
2 - B=sum(sum(A))
3 - C=prod(sum(A))
4

```

The **Command History** window shows the following commands:

```

B=sum(diag(A))
clc
A=[1 15 2 11; 23 1 4 5; 3 1 15 7; 1 4 9 10]
B=prod(diag(A))
clc
clear
clc
help magic
magic(3)
magic(9)
clc
A=magic(3)
B=magic(9)
clc

```

The **Command Window** shows the output of the commands:

```

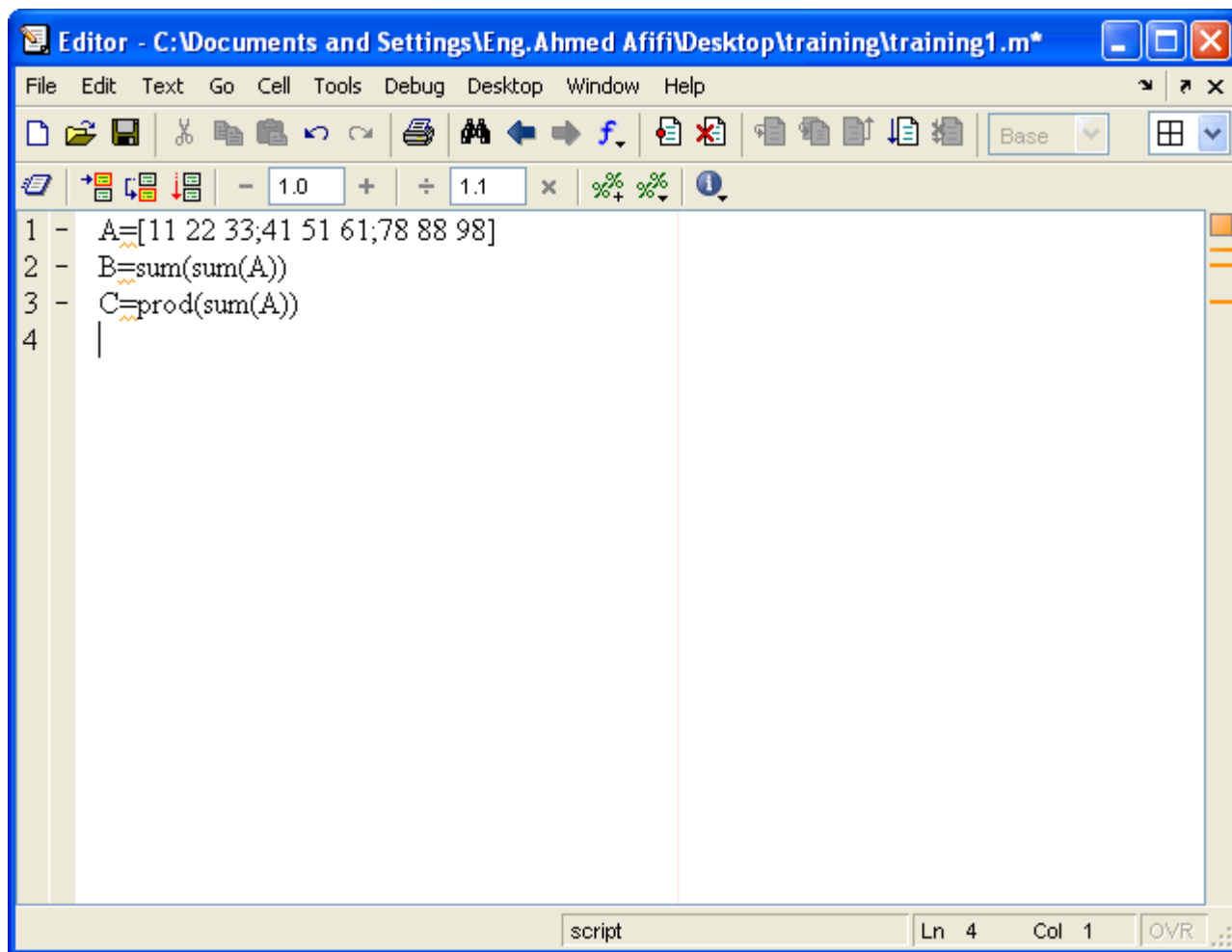
A =
     1     2     3
     4     5     6
     7     8     9

B =
    45

C =
   3240

```

4 - لنعود إلى M-File ونقوم بتغيير بعض القيم للمصفوفة, كما في الشكل التالي



The image shows a screenshot of a MATLAB script editor window. The window title is "Editor - C:\Documents and Settings\Eng.Ahmed Afifi\Desktop\training\training1.m*". The menu bar includes File, Edit, Text, Go, Cell, Tools, Debug, Desktop, Window, and Help. The toolbar contains various icons for file operations, editing, and execution. The script content is as follows:

```
1 - A=[11 22 33;41 51 61;78 88 98]
2 - B=sum(sum(A))
3 - C=prod(sum(A))
4 - |
```

The status bar at the bottom indicates the current position is "Ln 4 Col 1" and the file type is "script".

5 - سنقوم الآن بتشغيل البرنامج, وسيقوم الماتلاب الآن بالحفظ تلقائياً دون الحاجة لإعادة التسمية, ثم شاهد نافذة الأوامر Command Window

```
Command Window

1 2 3
4 5 6
7 8 9

B =
45
هذا قيم البرامج التي قد حصلنا عليها
منذ قليل

C =
3240

A =
11 22 33
41 51 61
78 88 98

B =
483
وهذه قيم البرنامج بعد عمل
التعديلات عليه

C =
4018560
```

وكما تلاحظ فإنه في كل عملية تحديث للبرنامج ستظل قيم البرنامج القديم موجودة, فحلاً لهذه المشكلة, يتم وضع الأمر `CLC` في أول كل برنامج, وهذا يكون مبدأ في جميع البرامج التي نقوم بعملها لابد من أن تبدأ بهذا الأمر, ودعونا نقوم بمثال يوضح لنا ذلك