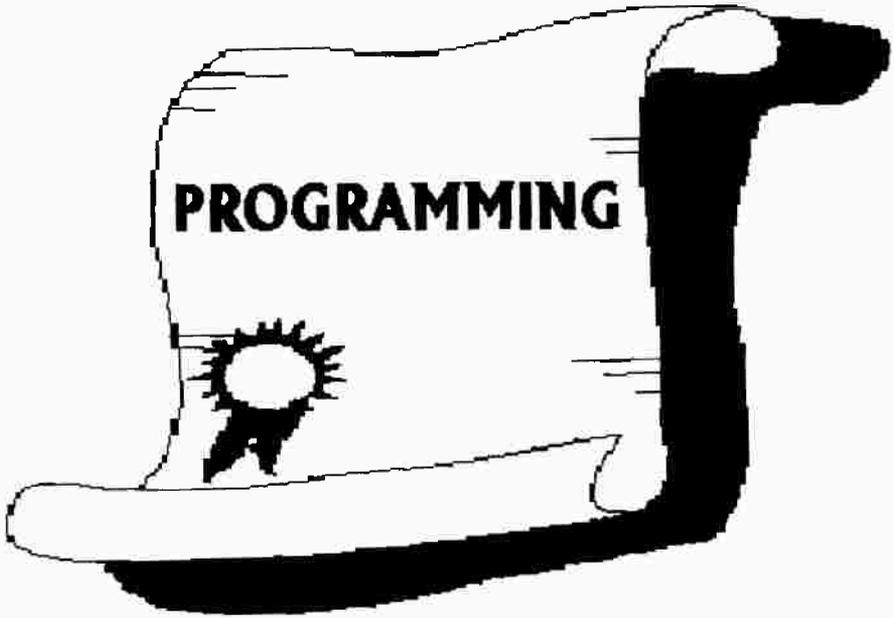


الباب السابع
البرمجة في ماثيماتيك



في هذا الباب سوف نتعرف على أوامر برنامج ماثيماتيك
والخاصة بالموضوعات الآتية :

Procedure

١. منظومة الإجراءات

Loops

٢. الحلقات التكرارية

Conditionals

٣. أوامر الانتقال المشروط

الباب السابع

البرمجة في ماثيماتيكا

عند بناء الحسابات في ماثيماتيكا غالبا ما نحتاج أن نربط مجموعة الأوامر معا ويمكن أن يتم ذلك باستخدام منظومة إجراءات Procedure .

١ . منظومة الإجراءات Procedure

منظومة الإجراءات هي عبارة عن متتابعة من أوامر أو تعبيرات ماثيماتيكا بحيث يفصل بين كلا منها علامة الفصلة المنقوطة ; وقيمة التعبير الأخير في المنظومة يمثل الناتج النهائي .

تعريف منظومة تتكون من مجموعة من الأوامر

Command1;Command2;...

In[1]:=r=(1+x)^2;r=Expand[r];r-1

Out[1]= 2 x + x²

في هذا المثال نلاحظ أن منظومة الإجراءات عبارة عن ثلاثة تعبيرات حامية يفصل كل منها عن الآخر بفصلة منقوطة كما نلاحظ أن الناتج النهائي هو قيمة التعبير الأخير (r - 1) في منظومة الإجراءات .

In[2]:= f[x_]:= (t=(1+x)^2;t=Expand[t]) ويمكن تعريف الدالة كمنظومة إجراءات
حيث نستخدم الأقواس () لتوضيح
أن كل الإجراءات معا تمثل الدالة f

In[3]:= f[a] حساب قيمة الدالة f عند x=a
Out[3]= 1+2a+a²

In[4]:= t عند الاستعلام عن قيمة t نجد أن المتغير
Out[4]= 1 + 2 a + a² t اصبح يمثل القيمة 1+2a+a²
ويحتفظ بها حتى بعد الخروج من المنظومة .

وفي كثير من الأحيان نحتاج الى أن نجعل المتغيرات المستخدمة داخل أى منظومة إجراءات كمتغيرات موضعية (Local) بمعنى أن هذه المتغيرات تحتفظ بالقيم داخل منظومة الإجراءات فقط ولكن تفقد هذه القيم بعد إنهاء الحسابات فى المنظومة والخروج منها ويتم عمل ذلك فى ماثماتيكيا باستخدام الأمر Block أو الأمر Module كالتالى :

<p>للإعلان عن أن المتغيرات x,y,... تمثل متغيرات موضعية داخل منظومة الإجراءات</p> <p style="text-align: right;">procedure</p> <p>Block[{x, y, ...}, procedure]</p> <p>Or</p> <p>Module[{x, y, ...}, procedure]</p>
<p>تحديد القيم الابتدائية ... y = yo , x = xo للمتغيرات الموضعية داخل منظومة الإجراءات</p> <p style="text-align: right;">procedure</p> <p>Block[{x=xo, y=yo, ...}, procedure]</p> <p>Or</p> <p>Module[{x = x0, y=yo, ...}, procedure]</p>

وبذلك فإنه بواسطة الأمر **Block** أو الأمر **Module** يمكن تعريف أى متغيرات داخل منظومة الإجراءات الواحدة دون التأثير على القيم بخارج المنظومة وبالتالي يمكن تعريف نفس المتغير الموضعى داخل اكثر من منظومة إجراءات .

In[5]:= عند استخدام الأمر **Block** فى تعريف

g[x_]:=Block[{u},u=(1+x)^2;u=Expand[u]] الدالة **g** فان **u** يعامل كمتغير موضعى

In[6]:= g[a] لحساب قيمة الدالة **g** عند **x = a**

Out[6]= 1 + 2a + a²

In[7]:=u عند الاستعلام عن قيمة **u** نلاحظ عدم

Out[7]=u وجود قيمة لأن **u** متغير موضعى

In[8]:=x=5; تعريف **x=5** خارج المنظومة ثم استخدام

Module[{x},x=Random[];Print[x];] الأمر **Module** واعتبار أن **x** متغير موضعى

Out[8]=0.863718 وطباعة قيمة عشوائية للمتغير **x**

In[9]:=?x عند الاستعلام عن قيمة **x** نلاحظ أن

Out[9]=Global`x **x = 5** وهى القيمة الموجودة خارج

x = 5 الأمر **Module**

٢ . الحلقات التكرارية Loops

منظومة الإجراءات تسمح بتنفيذ مجموعة من تعبيرات ماتيماتيكا وفقا لترتيب هذه التعبيرات داخل منظومة الإجراءات وفي كثير من الأحيان نحتاج الى تنفيذ بعض العمليات بصورة متكررة ويتم ذلك داخل ماتيماتيكا باستخدام أوامر خاصة بالحلقات المتكررة Loops حيث تعمل الحلقات المتكررة على تكرار مجموعة متتالية من الأوامر بصورة متكررة لعدد محدود من المرات مع إمكانية التغيير الأوتوماتيكي لقيم المتغيرات داخل الحلقات التكرارية ، ومن أوامر ماتيماتيكا الخاصة بالحلقات التكرارية هو الأمر Do والذي يستخدم بصورة مشابهة كما فى لغات البرمجة مثل فورتران FORTRAN والأمر Do له استخدامات متعددة وصيغته العامة موضحة بالجدول الآتى :

الصيغة العامة للأمر	الوظيفة التى يقوم بها الأمر
Do[expr,{n}]	حساب قيمة expr عدد n من المرات
Do[expr,{i,imax}]	حساب قيمة expr بصورة متكررة وفقا للعداد i من i=1 الى i=imax بخطوة تساوى 1
Do[expr,{i,imin,imax}]	حساب قيمة expr بصورة متكررة وفقا للعداد i من i=imin الى i=imax بخطوة تساوى 1
Do[expr,{i,imin,imax,istep}]	حساب قيمة expr بصورة متكررة وفقا للعداد i من i=imin الى i=imax بخطوة تساوى istep
Do[expr,{i,imin,imax}, {j,jmin,jmax}]	حساب قيمة expr لقيم i, j المعطاة

In[1]:=t=x;Do[t=2(1+t),{3}];t
Out[1]=2 (1 + 2 (1 + 2 (1 + x)))

في هذه الحلقة يتم حساب التعبير
 $t=2(1+t)$ ثلاث مرات حيث $t=x$

In[2]:=Do[Print[m^2],{m,3}]
Out[2]= 1
4
9

في هذه الحلقة يتم طباعة m^2 لقيم
m من 1 الى 3

In[3]:=Do[a=i^2+3j;Print[a],{I,2},{j,I}]
Out[3]= 4
7
10

في هذه الحلقة يتم حساب التعبير
 $a = i^2 + 3j$ ثم طباعته لقيم
 i, j المعطاة

وفي برنامج ماثماتيكا يمكن تكرار تطبيق نفس الدالة عدد محدود من المرات على تعبير معين باستخدام الأمر Nest كالآتي :

Nest[f,expr,n]

تطبيق الدالة f على التعبير expr
لعدد n من المرات

In[4]:=Nest[f,x,3]
Out[4]= f[f[f[x]]]

لحساب $f(f(f(x)))$

In[5]:=f[x_]=(x+1)^2;Nest[f,x,2]
Out[5]=(1 + (1 + x)^2)^2

تعريف الدالة $f(x) = (x+1)^2$
ثم حساب $f(f(x))$

In[6]:= Nest[f,1,2]
Out[6]= 25

لحساب $f(f(1))$

وفي برنامج ماتيماتيكا يمكن بناء الحلقات بحيث يتم إيقاف تنفيذ التكرار إذا لم يتحقق شرط معين وذلك باستخدام الأوامر **For , While** كالتالي :

الصيغة العامة للأمر	الوظيفة التي يقوم بها الأمر
For[start,test,step,body]	حساب قيمة start والتحقق من الشرط test لتنفيذ body مع تكرار إضافة step
While[test,body]	يتم تكرار تنفيذ body إذا كان الشرط test متحقق

In[7]:=For[i=0,i<3,i=i+1,Print[i]]

حلقة باستخدام **For** حيث يتم البدء

Out[7]=

بقيمة **i=0** والتحقق من الشرط **i<3**

0

لتنفيذ طباعة **i** مع إضافة 1 الى **i**

1

2

In[8]:=i=0;While[i<3,Print[i];i=i+1]

تنفيذ الحلقة السابقة باستخدام **While**

Out[8]=

0

1

2

وعند استخدام أوامر الحلقات فى ماتيماتكا خاصة مع الأوامر For , While غالباً ما نحتاج الى تكرار تعديل قيم فى بعض المتغيرات المستخدمة داخل الحلقة ، وتوجد بعض الطرق المختصرة لأجراء مثل هذه التعديلات فى قيم المتغيرات والجدول الآتى يوضح ذلك .

العملية المختصرة	معنى العملية المختصرة
$i++$	زيادة قيمة i بمقدار 1 فيما يستجدمع الاحتفاظ بقيمة i السابقة داخل $i++$
$i--$	نقصان قيمة i بمقدار 1 فيما يستجدمع الاحتفاظ بقيمة i السابقة داخل $i--$
$++i$	زيادة قيمة i بمقدار 1 وجعل i هى القيمة الجديدة أى أن $++i$ تمثل $i+1$
$--i$	نقصان قيمة i بمقدار 1 وجعل i هى القيمة الجديدة أى أن $--i$ تمثل $i-1$
$i+ = d$	زيادة قيمة i بمقدار d أى أن $i=i+d$
$i- = d$	نقصان قيمة i بمقدار d أى أن $i=i-d$
$x*=c$	ضرب x فى العدد c أى أن $x = x * c$
$x/=c$	قسمة x على العدد c أى أن $x = x / c$
$\{x,y\}=\{y,x\}$	استبدال قيم x, y أى تغيير قيمة x لتصبح y وتغيير قيمة y لتصبح x

In[9]:=i=5;Print[i++];Print[i]

Out[9]=

5

6

في هذا المثال يتم طباعة قيمة $i++$ وقيمة i ونلاحظ أن قيمة $i++$ هي قيمة i قبل الزيادة

In[10]:=i=5;Print[++i];Print[i]

Out[10]=

6

6

في هذا المثال يتم طباعة قيمة $++i$ وقيمة i ونلاحظ أن قيمة $++i$ هي قيمة i بعد الزيادة

In[11]:=r=x;r+=3y;r

Out[11]=x + 3 y

في هذه المنظومة تم وضع $r=x$ ثم زيادة قيمة r بمقدار $3y$ فتصبح قيمة r الجديدة $x+3y$

In[12]:=a=3;b=7;Print[{a,b}];

{a,b}={b,a};Print[{a,b}]

Out[12]=

{3, 7}

{7, 3}

استبدال قيم a, b

In[13]:=x=1;y=2;z=3;Print[{x,y,z}];

{x,y,z}={z,x,y};Print[{x,y,z}]

Out[13]=

{1, 2, 3}

{3, 1, 2}

استبدال قيم x, y, z

In[14]:=For[i=1;t=x,i^2<10,i++,t=t^2+i;Print[t]]

Out[14]=

$$1+x^2$$

$$2+(1+x^2)^2$$

$$3+(2+(1+x^2)^2)^2$$

هذه الحلقة على الصورة **For[start,test,step,body]** حيث

start	i=1 ; t=x
test	i^2<10
step	i++
body	t=t^2+i ; Print[t]

٣ . أوامر الانتقال المشروط Conditionals

عند بناء منظومة الإجراءات Procedure في ماتيماتكا غالبا ما نحتاج الى تنفيذ بعض العمليات إذا تحقق شروط معينة ويتم ذلك في ماتيماتكا باستخدام أوامر الانتقال المشروط الآتية :

الصيغة العامة للأمر	الوظيفة التي يقوم بها الأمر
If[test,then,else]	يتم تنفيذ then إذا كان test تحقق وخلاف ذلك يتم تنفيذ else
If[test,then,else,unknown]	إذا كان test تحقق يتم تنفيذ then وإذا كان test غير متحقق يتم تنفيذ else وخلاف ذلك يتم تنفيذ unknown
1,value1,test2,value 2, ...]	يتم تنفيذ value المناظرة الى أول اختبار testi يتحقق

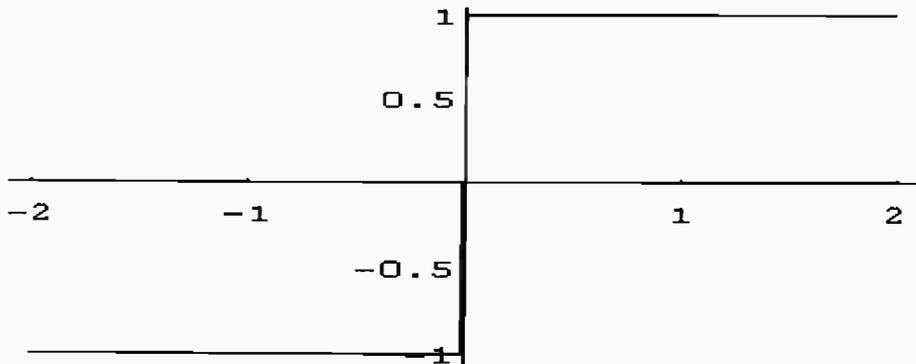
```
In[1]:= x=3;y=5;
        If[ x > y , Print[x] , Print[y] ]
```

```
Out[1]=
        5
```

إدخال قيم x, y ثم طباعة العدد الأكبر وقد تم استخدام أمر الانتقال المشروط **If** على الصورة **If[test,then,else]** حيث

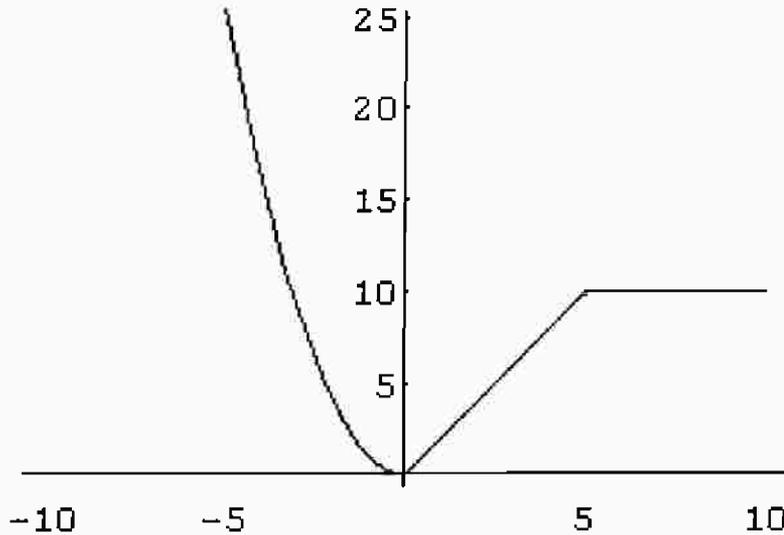
test	$x > y$
then	Print[x]
else	Print[y]

```
In[2]:=f[x_]:=If[x>0,1,-1];
        Plot[f[x],{x,-2,2}]
```



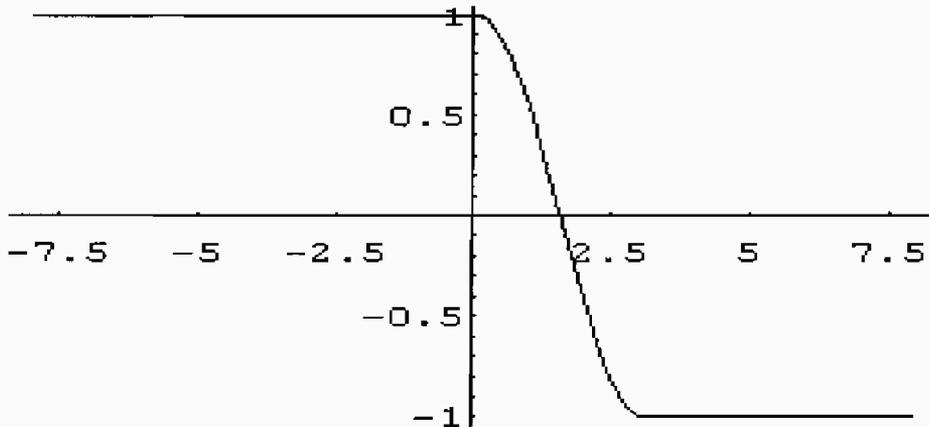
استخدام الأمر **If** في تعريف الدالة $f[x]$ ثم رسم الدالة

```
In[3]:=r1[x_]:=Which[x<0,x^2,x>0 && x<5,2x,True,10];
Plot[r1[x],{x,-10,10}]
```



استخدام الأمر Which في تعريف الدالة r1[x] ثم رسم الدالة

```
In[4]:=r2[x_]:=Which[x<0,1,x<N[Pi],Cos[x],True,-1];
Plot[r2[x],{x,-8,8}]
```



استخدام الأمر Which في تعريف الدالة r2[x] ثم رسم الدالة

منظومة تم فيها تعريف قائمة a من الأعداد الحقيقية ثم حساب وطباعة العدد الأصغر من القائمة

```
In[5]:=a={5,2,7,55,-4,9,3,10,-24,44,65,-21};mi=a[[1]];
Do[If[mi>a[[i]],mi=a[[i]],++i],{i,1,Length[a]-1}];
Print["Minimum of the list a = ",mi]
```

Out[5]= Minimum of the list a = -24

منظومة تم فيها حساب وطباعة العدد الأكبر من القائمة a

```
In[6]:= ma=a[[1]];
Do[If[ma<a[[i]],ma=a[[i]],++i],{i,1,Length[a]-1}];
Print["Maximum of the list a = ",ma]
```

Out[6]= Maximum of the list a = 65

منظومة تم فيها حساب وطباعة العدد الأصغر والعدد الأكبر من القائمة a

```
In[7]:= mi=a[[1]];ma=a[[1]];
Do[Which[mi>a[[i]],mi=a[[i]],ma<a[[i]],ma=a[[i]]];++i,
{i,2,Length[a]-1}];
Print["Minimum of the list a = ",mi];
Print["Maximum of the list a = ",ma]
```

Out[7]= Minimum of the list a = -24
Maximum of the list a = 65

وباستخدام الدوال الموجودة داخل بناء ماثماتیکا يمكن حساب

Min العدد الأصغر من القائمة مباشرة باستخدام الدالة

Max والعدد الأكبر من القائمة مباشرة باستخدام الدالة

```
In[8]:=Min[a]
Out[8]=-24
```

```
In[9]:=Max[a]
Out[9]=65
```


المراجع

- [1] - **Wolfram , Stephen**
Mathematica : A System for Doing Mathematics
by Computer ,
Second Edition , Addison Wesley , 1991 .
- [2] - **Wolfram , Stephen**
Mathematica : The Student Book ,
Addison Wesley , 1994 .
- [3] - **Abell , Martha L . and Braselton, Tames P . ,**
The Mathematica Handbook
Academic Press , 1992 .
- [4] - **Maeder , Roman ,**
Programming in Mathematica ,
Addison Wesley , 1992 .