كلية العلوم

القسم : الفيزياء

السنة : الثانية

المادة : لغات البرمجة2

المحاضرة : الثالثة /ن+ع/

{{ مكتبة A to Z }}

Facebook Group : مكتبة A to Z

المحاضرة الثالثة Scripts vs. Function

تعلمنا فيما سبق أنه عند النهاية من كتابة سطر برمجي في نافذة الأمر، فإنه مباشرة يتم تنفيذه ويخزن قيمته في نافذة العمل.

لذلك تم معالجة هذه المشكلة عن طريق كتابة البرامج ضمن ملفات وتخزينها في الماتلاب بامتداد(m)، وهذه الملفات هي ملفات نصية من نوع (ASCII) ويمكن التحكم فيها بمرونة وتصحيحها، كما يمكن إضافة برامج فرعية لها ويتم تنفيذها في الوقت المناسب (أي انها مرنة).

- الامر :What يظهر الملفات الفعالة. >>what

M-File يمكن أن تكون ملفات نصية (script file) أو دوال (function file).

كلا النوعين السابقين تتضمن سلسلة متتابعة من الأوامر ولكن ((**function file**. يمكن أن يسند لها قيم وتعطي النتائج...وسوف نوضحها من خلال الأمثلة.

1. **script file:**

هذا الملف النصي مفيد جدا للتحليل وإيجاد حلول المسائل التي تتطلب سلسلة طويلة من العمليات، ويتم عرض النتائج في نافذة الامر بعد إعطاء امر التنفيذ، حيث يبين أيضا الأخطاء الواردة في البرنامج عند وجودها:

عند كتابة البرنامج يجب تخزينه باسم معين ويتم استدعائه بنفس الاسم عند الطلب:

```
X=input ('Enter the number x :');
if (X>=0)
disp ('The number is positive')
Else
disp ('The number is positive')
End
```

عند التنفيذ في نافذة الأمر:

```
>> phizic111
Enter the number x: 12
The number is positive
```

مثال2:

```
x=input('enter x=');
if(x>=0)
    y=x+100;
else
    y=x^2-6;
end
disp('result=');
disp(y)
```

نخزن البرنامج باسم (ph2)

ويتم التنفيذ عند القيمتين: x=10, x=-100

```
>> ph2
Enter x=10
Result=
110
>> ph2
Enter x=-100
Result=
 9994
```

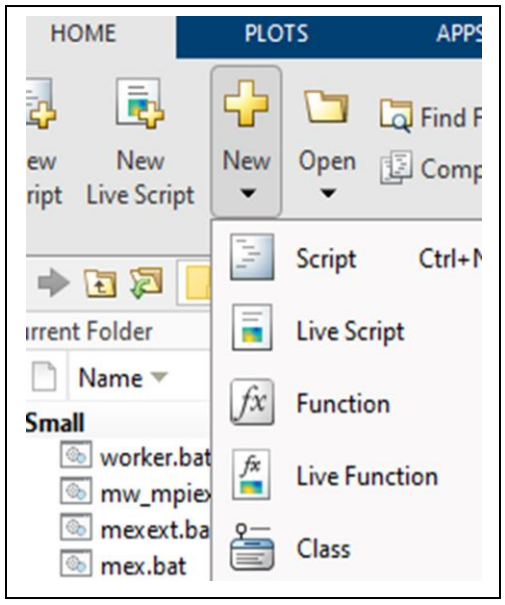
مثال 3: للحل:

ادخل رقما، وطبق على الشروط المنطقية والعلائقية: مثلا إذا كان أكبر او يساوي 5، قم بالعملية الأولى وإذا كان الصفر والخمسة، اجري العملية الثانية، وإذا كان أصغر من الخمسة خم بتربيعه ثم اعرض النتيجة؟

نقوم بتخزين البرنامج(ph3)، ونعطي قيم المتحول: x=8,4,-5 تكون النتائج في نافذة الامر:

```
>> ph3
Enter x=8
108
>> ph3
Enter x=4
 17
>> ph3
Enter x=-5
 -125
```

Scripts vs. Function ملفات الدوال



تستخدم هذه الملفات لإنشاء توابع جديدة غير مشهورة في الماتلاب، العمليات والمتغيرات في الدالة هي عبارة عن متحولات خاصة بالدالة. الشكل المجاور:

عند كتابة أي برنامج يمكن أن تكون طريقة البرمجة بطريقتين وهما إما أن يكتب البرنامج بشكل مباشر وهذه الطريقة تسمى **(script)** أو أن ننشئ تابع ونضع ضمنه التعليمات البرمجية اللازمة لتنفيذ البرنامج وهذه الطريقة **(function)** وهناك عدة فروق بين هاتين الطريقتين:

1. الـ function يبدأ بكلمة function بينما الـ script يبدأ بالبرنامج مباشرة.
2. الشكل الافتراضي للكتابة بعد كلمة function هو أن تضع وسطاء الخرج ومن ثم اسم الـ function وبعدها وسطاء الدخل، بينما الـ scripts لا يحتاج إلى كل هذا.
3. يمكنك التابع (function) عند تنفيذه من تغيير وسطاء الدخل بينما لا يكون هذا ممكن في الـ scripts.
4. المتغيرات التي تحسب في داخل الـ function لا تحفظ في الـ workspace بينما في الـ scripts تحفظ في الـ workspace (أي في لوحة الدخل والخرج الرئيسية لبرنامج الماتلاب).

والصيغة العامة للدوال: Function variable(s) =function name (arguments)

حل معادلة من الدرجة الثانية بطريقتين:

1. Function:

```
function vals1(a,b,c)
delta = b^2-4*a*c
if delta>0
x1=(-b+sqrt(delta))/(2*a)
x2=(-b-sqrt(delta))/(2*a)
```

```
elseif delta<0
disp('the roots are complex')
else
x1_2= (-b/(2*a))
end
```

النتائج: يتم طلب التنفيذ من نافذة الامر بعد تخزين اسم التابع بنفس اسمه، مثلا: Vals1

```
>>vals1(1,2,1)
delta =0  , x1_2 =-1  ,
>>vals1(1,2,5)
delta = -16 , the roots are complex
>>vals1(1,5,1)
delta =21  , x1 =-0.2087    ,   x2 =4.7913-
```

2. **Script**:

```
a=input('a=');
b=input('b=');
c=input('c=');
delta = b.^2-4*a*c
if delta>0
x1=(-b+sqrt(delta))/(2*a)
x2=(-b-sqrt(delta))/(2*a)
elseif delta<0
disp('the roots are complex')
else
x1_2= (-b/(2*a))
end
```

يتم اعطاء امر التنفيذ بعد تخزينه من نافذة (script) وتظهر النتائج في نافذة الامر:

```
>> ph4
a=1
b=2
c=1
delta =     0,   x1_2 =-1
```

العمليات على كثيرات الحدود

تمثل كثيرات الحدود في الماتلاب بشكل نسق من العناصر (الأعداد)، وهذه الأعداد هي المعاملات العددية للمتحول (x) ابتداء من القوة (n) إلى القوة (0).يجب تمثيل جميع المعاملات العددية حتى المعدومة منها.

$$p = 7x^3 + 6x^2 - 2x + 100$$

```
>>p= [7 6 -2 100];
```

مثلا: أوجد جذور كثير الحدود التالي:

$$p1 = x^4 - 10x^3 + 50x^2 - 30x - 15$$

```
>>p1=[ 1 -10 50 -30 -15];
>>roots_p1= roots(p1)
roots_p1 =
                    4.6163 + 4.6852i
                    4.6163 - 4.6852i
                    1.0866 + 0.0000i
                     -0.3191 + 0.0000i
```

كيف يتم بناء كثير الحدود انطلاقا من جذوره:

يمكن معرفة معاملات كثير الحدود عن طرق استخدام التعليمة ((poly(r))، حيث(r) نسق يتضمن جذور كثير الحدود.

تمرين:

أوجد أمثال كثير الحدود له الجذور الآتية:

```
R=-1, -2 , -3, 4+j5,4-j5
>> R=[-1 -2 -3 4+j*5 4-4*j];
>>poly_R=poly(R)
poly_R =
 1   -2    4   164   403   246
```

يكون كثير الحدود المطلوب:

$$x^5 - 2x^4 + 4x^3 + 164x^2 + 403x = -246$$

كيفية حساب كثير الحدود عند قيمة محددة للمتحول:

يتم ذلك عن طريق استخدام الدالة: (polyval) الذي يقبل وسيطين: الأول هو النسق الذي يمثل لمعاملات كثير الحدود، والثاني هو قيمة المتحول (x) أو نسق القيم المراد إسنادها إلى المتحول (x)، وتأخذ التعليمة الصيغة التالية:

```
Fx=polyval(p,x)
```

مثال:

المطلوب إيجاد قيمة كثير الحدود التالي:

$$p(x) = x^6 - 3x^5 + 5x^3 - 4x^2 + 3x + 2$$

1. X=-3:

```
>> P=[1 -3  5  -4  3  2];
>> p3=polyval(P,-3)
p3 =  -664
```

2. X=[1 2 -2] :

```
>>px=polyval(p,x)
px =4   16  -140
```

هناك دوال أخرى في الماتلاب لكثيرات الحدود:

العملية	التمثيل
الجمع	+
الطرح	-
جداء كثيري الحدود	Conv(y,z)
قسمة كثيري الحدود	Deconv(y,z)
تكامل كثير الحدود	Polyint(p)

الدوال الرياضية الهامة في الماتلاب:

الشرح	الدالة
القيمة المطلقة	Abs(x)
تقريب الرقم العشري أو المصفوفة باتجاه (∞)	Ceile(x)
جيب تمام الزاوية(x)	Cos(x)
المرفوع لقوة بأساس عشرة	Exp(x)
تقريب الرقم العشري باتجاه الصفر، الغاء الكسر والحصول على الرقم الصحيح فقط	Fix(x)
تقريب الرقم العشري أو المصفوفة باتجاه (−∞)	Floor(x)

Image(x)	العدد التخيلي بالعدد العقدي
log(x)	اللوغاريتم الطبيعي
log10(x)	اللوغاريتم العشري
real(x)	العدد الحقيقي بالعدد العقدي
Rem(x,y)	اخراج الباقي الصحيح لعملية القسمة >>rem(5,2) ans = 1
Round(x)	تقريب الرقم العشري باتجاه أقرب رقم الصحيح >>round(5.235) ans = 5
Sin(x)	حساب جيب الزاوية << xsin(30/180) *pi = 0.5212
Sqrt(x)	إيجاد الجذر التربيعي للعدد x
Tan(x)	حساب ظل الزاوية x
Pi	Pi=3.14
Ans	قيمة العنصر الناتج من إسناد القيمة
Ged	القاسم المشترك الأكبر
Lem	المضاعف المشترك الأصغر
primes	ينشىء قائمة بالأعداد الأولية
min	العنصر الأصغر من المتجه أو المصفوفة
mod	الجزء الصحيح من حاصل القسمة
Log2	اللوغاريتم ذو الأساس 2

نهاية المحاضرة

المحاضرة العملية

اكتب برنامجا تنشئ فيه تابعا اسمه (sort) بحيث يقوم بترتيب شعاع من العناصر ترتيبا تنازليا:

```
function g=sort(a)
s=length(a);
for i=1:s-1
for j=i+1:s
if a(i)<a(j)
x=a(i);
a(i)=a(j);
a(j)=x;
end
end
end
a
```

نقوم بتنفيذ البرنامج من نافذة الامر:

```
>> a= [15 -9 89 10 23 45 65];

a =

89  65  45  23  15  10  -9
```

مثال2: أما اذا كانت عناصر المصفوفة معلومة:

```
matlab
% مثال: ترتيب مصفوفة أحادية من 15 عنصراً ترتيباً تنازلياً
% بطريقة مشابهة للكود المرفق

% تعريف المصفوفة الأولية (أعداد صحيحة عشوائية)
a = [5, 12, 3, 8, 15, 1, 9, 20, 7, 14, 2, 18, 11, 6, 4];

% عرض المصفوفة قبل الترتيب
disp('المصفوفة قبل الترتيب:');
disp(a);

% خوارزمية الترتيب التنازلي (Bubble Sort - Descending)
s = length(a);
for i = 1:s-1
    for j = i+1:s
```

```
        if a(i) < a(j)   % إذا كان العنصر الأول أصغر من الثاني
            x = a(i);    % نقوم بالتبديل (للترتيب التنازلي)
            a(i) = a(j);
            a(j) = x;
        end
    end
end

% عرض المصفوفة بعد الترتيب
disp('المصفوفة بعد الترتيب التنازلي:');
disp(a);
```

مثال 3: اوجد اطوال اضلاع مثلث بعد معرفة زوياه مستخدما قالب الدالة:

```
function triangle_sides()
A=input('Angle A (deg): ');
 B=input('Angle B (deg): ');
 C=input('Angle C (deg): ');
if abs(A+B+C-180)>0.01 disp('Sum error'); return; end
disp('Known side: 1=a(opp A) 2=b(opp B) 3=c(opp C)');
ch=input('Choice: ');
ks=input('Length: ');
A=A*pi/180; B=B*pi/180; C=C*pi/180;
switch ch
   case 1; r=ks/sin(A); a=ks; b=r*sin(B); c=r*sin(C);
   case 2; r=ks/sin(B); a=r*sin(A); b=ks; c=r*sin(C);
   case 3; r=ks/sin(C); a=r*sin(A); b=r*sin(B); c=ks;
end
disp(['a=' num2str(a) ' b=' num2str(b) ' c=' num2str(c)]);
end
```

أمثلة عن كثيرات الحدود:

```
% جمع كثيرات الحدود
p1 = [1 3 2];
p2 = [1 0 -1];
p_sum = p1 + p2
% طرح كثيرات الحدود
p_sub = p1 - p2
% ضرب كثيرات الحدود
p3 = [1 2];
p4 = [1 -2];
p_mul = conv(p3, p4)
% قسمة كثيرات الحدود
p5 = [1 0 -4];
p6 = [1 2];
[Q, R] = deconv(p5, p6)

% جذور كثير الحدود
p7 = [1 0 -4];
r = roots(p7)

% بناء كثير الحدود من جذوره
roots_vec = [2, -2, 3];
p8 = poly(roots_vec)

% قيمة كثير الحدود عند نقطة
p9 = [1 -2 1];
val = polyval(p9, 5)

% قيمة عند عدة نقاط
x_vals = [0 1 2 3];
y_vals = polyval(p9, x_vals)
```

```
% تكامل كثير الحدود
p10 = [2 3 1];
p_int = polyint(p10)

% اشتقاق كثير الحدود
p11 = [1 3 2];
p_der = polyder(p11)
% جمع كثيرات حدود بأطوال مختلفة
p12 = [1 0 5];
p13 = [1 1];
p12_padded = [0 p12];
p_sum2 = p12_padded + p13
% ضرب 3 كثيرات حدود
p14 = [1 0];
p15 = [1 1];
p16 = [1 -1];
p_mul3 = conv(conv(p14, p15), p16)
% تقييم كثير الحدود باستخدام polyvalm
p17 = [1 -2 1];
M = [1 2; 3 4];
result = polyvalm(p17, M)
% استخراج المعاملات من جذور مركبة
roots_c = [1+j, 1-j, -2];
p18 = poly(roots_c)
% ضرب وقسمة متتالية
p19 = [1 4 4];
p20 = [1 2];
p21 = conv(p19, p20)
p22 = [1 6 12 8];
[Q2, R2] = deconv(p22, p20)
**********
```

نهاية المحاضرة العملية