Image processing with MATLAB

Basic Programming in MATLAB

Setting By : L. Waleed Rasheed

Third Lecture

البرمجة الاساسية في ماتلاب

يوفر برنامج ماتلاب امكانية كتابة مجموعة من الاوامر والايعازات من خلال ملف تحرير الـ M-file الذي يمكن الوصول له عن طريق File→New→Script او بضغط مفتاحي (Ctrl+N) حيث يتيح لنا هذا الملف كتابة جميع الاوامر وتنفيذها دفعة واحدة من خلال اختيار قائمة Debug→Run او من خلال تحديد الكل باختيار Edit→Select All ومن تقييم التنفيذ من خلال Text→Evaluate Selection حيث سيظهر التنفيذ او الخطأ ان وجد مع رقم السطر الذي يحتوي على الخطأ.

ويمكن اضافة تعليقات او ملاحظات داخل الملف من خلال كتابة سطر نصي كامل مسبوق بعلامة (%).

مثال : اكتب برنامج في لغة ماتلاب باستخدام M-filesلايجاد ناتج المعادلة التالية ؟y=a*b+c

```
a=5; b=6; c=12;
y=a*b+c
```

أوامر الادخال والاخراج

الامر input: يستخدم الامر input الادخال الارقام بشكلها الصحيح او الكسور العشرية ، ويكتب الامر بالشكل التالي.

```
x=input('enter x No.')
```

كما يمكن من خلاله ادخال السلاسل الحرفية بعد اضافة 's' الى صيغة الامر للدلالة على ان المدخل هو سلسلة حرفية، كما موضح في الصيغة التالية.

```
y =input('enter your name','s')
```

الامر disp: يستخدم لعرض النتائج بشكل مباشر عبر وضع اسم المتغير بعد الامر وبين قوسين، ويكتب بالشكل التالي.

```
disp(variable name)
EX// disp(x)
```

ويمكن عرض النصوص الحرفية باستخدام نفس الامر وبالشكل التالي

```
EX// disp('waleed')
```

أوامر الادخال والاخراج

الامر fprintf: يستخدم هذا الامر للتحكم في شكل الطباعة وعدد مراتب بعد تحديد نوع المخرجات على الشاشة، وكما موضح في الامثلة التالية.

```
a = [1.02 3.04 5.06];
fprintf('%d\n', round(a));
```

```
Output
1
3
5
```

حيث ان %d ترمز الى ان الرقم المطبوع عشري صحيح ، وان \n تمثل ان كل رقم سيتم طباعته على سطر جديد، وهنا يمكن ان نذكر عدد من الترميزات المفيدة مع ايعاز fprintf ومايقابلها من تمثيل في المخرجات.

%c	رمز واحد	%f	كسر عشري	%s	سلسلة حرفية	%x	نظام السادس عشر

ويمكن لهذا الامر ان يحدد عدد المراتب العشرية بعد الفارزة ايضا، كما موضح في المثال التالي.

```
>> x=rand(2)
x =
   0.9649   0.9706
   0.1576   0.9572
```

```
>> fprintf('variable x is % 6.2f\n',x);
variable x is   0.96
variable x is   0.16
variable x is   0.97
variable x is   0.96
```

أوامر الادخال والاخراج

ويمكن لإيعاز fprintf ان يحول المخرجات الى ملف نصي يتم خزنه على الحاسوب بعد تحديد المسار واستخدام الاوامر fopen لفتح الملف و fclose لغلق الملف

مثال : اكتب برنامج لبناء جدول بسيط يوضح قيم دالة exp ليتم عرضه على الشاشة وحفظه في ملف نصي بالاسم exp.txt؟

```
x = 0:.1:1;
y = [x; exp(x)];
fprintf('%6.2f %12.8f\n', y);

% open the file with write permission
fid = fopen('c:\exp.txt', 'w');
fprintf(fid, '%6.2f %12.8f\n', y);
fclose(fid);
```

```
0.00   1.00000000
 0.10   1.10517092
 0.20   1.22140276
 0.30   1.34985881
 0.40   1.49182470
 0.50   1.64872127
 0.60   1.82211880
 0.70   2.01375271
 0.80   2.22554093
 0.90   2.45960311
 1.00   2.71828183
```

الجمل الشرطية

برنامج ماتلاب يدعم بناء الجمل الشرطية بانواعها المختلفة ويستخدم معها المشغلات المنطقية والعلائقية التي تم شرحها مسبقاً، ويسمح ببناء جمل شرطية بسيطة او معقدة ، وان ناتج كل مقارنة تحدث هي صحيح True او خطأ False وعلى ضوء ناتج المقارنة يتم تنفيذ الخطوات التابعة لها.

الامر If Statement : ان الصيغة الاساسية لاستخدام الامر if في ماتلاب هي .

If condition **Statement** **End**	**If condition** **Statement1** **Else** **Statement2** **end**
Exmple1 :	**Exmple2 :**

if(a>=50)
p=1
else
p=2
end
	if(deg>=50)
	disp('succ.')
	else
	disp('faild')
	end

الجمل الشرطية

ويمكن بناء جمل شرطية متداخلة Nested If ، مثلا اذا كان لديك الرقم x والمطلوب فحص الرقم اذا كان موجب او سالب او صفر، تكتب الجملة بالشكل التالي

```
if (x>0)disp('Pos.')
else if (x<0)disp('Neg.')
else disp ('zero')
end
end
```

حيث يتم وضع end لكل عبارة if، وفي حالة كتابة if ملاصقة لكلمة else في الشروط التالية يمكن استخدام endواحدة فقط، حيث يتم تعديل صيغة المثال السابق ويكتب بالشكل التالي

```
if (x>0)disp('Pos.')
elseif (x<0)disp('Neg.')
else disp ('zero')
end
```

ويمكن بناء جمل شرطية مركبة بالاعتماد على مجموعة العلاقات المنطقية ، وكما يلي

```
if (deg1>=70 && deg2>=70)
disp('good')
end
```

جملة Switch-Case

استخدم هذه الجملة يقلص تكرار استخدام if للسؤال عن عنصر معين سواء كان رقم او رمز او سلسلة حرفية، ويكتب بالصيغة التالية

```
switch switch_expr
case case_expr
      statement,...,statement
case {case_expr1,case_expr2,...}
      statement,...,statement...
otherwise
statement,...,statement
end
```

Simple example

```
----------
switch n
case 1
   disp('first')
case {2,3}
  disp('Second or third')
case 4
   disp('fourth')
otherwise
   disp('Last')
end
```

في المثال السابق سيتم البحث عن قيمة n في الحالات الموجودة ومن ثم طباعة الجملة اللاحقة للحالة الصحيحة وفي حالة عدم وجود اي حالة مطابقة سيتم تنفيذ الجملة التي بعد otherwise.

الجمل التكرارية

هي جمل تستخدم لتكرار مجموعة من الخطوات البرمجية لعدد محدد من المرات، ومنها .

1-جملة for:

ان الصيغة العامة لكتابة for هي

For = first_value : last_value **Statement1** **...** **Statement_n** **end**	**Simple Example** **----------** **for i=1:5** **x(i)=i;** **end**	**>> x** **x =** **1 2 3 4 5**

ويمكن التحكم بمقدار الزيادة والنقصان من خلال اضافة رقم بين القيمة الاولى والاخيرة في جملة for، كما موضح في المثال التالي.

for i=2:2:10 **disp(i)** **end**	**2** **4** **6** **8** **10**

الجمل التكرارية

2-جملة while: تكتب الصيغة العامة لجملة while بالشكل التالي..

Form:	Example	Output
--------	--------	--------

While condition
Statement
Counter
end
	i=1;
	while(i<3)
	disp(i)
	i=i+1;
	end
		1
		2

تستمر جملة while بالتنفيذ في حالة الشرط الصحيح ويتوقف التنفيذ في حالة تحول الشرط الى خطأ.

التكرارات المتداخلة Nested Loop مدعومة ايضا في لغة ماتلاب واليك المثال التالي لتوضيح طريقة كتابة التكرارات المتداخلة.

```
for i=1:3
  for j=1:3
    y(i,j)=i+j;
  end
end
```

```
>> y
y =
     2     3     4
     3     4     5
     4     5     6
```

كتابة الدوال والبرامج الفرعية في ماتلاب

ان البرامج الفرعية جزء لا يتجزء من البرمجة الاساسية في اي لغة برمجية لقدرتها على تجزئة المشكلة الكبيرة الى مجموعة من المشاكل الفرعية التي يسهل وضع الحلول لها ومن جانب اخر تسهل عملية تشخيص الخطأ ان وجد أثناء عملية كتابة الشفرة البرمجية.

وهنا نذكر الضيغة العامة لكتابة الدالة

Function [var1_out,var2_out,…]=function_name(var1_in,var2_in,…)

حيث ان : var1_out تمثل المتغير (أو اكثر) الناتج عن تنفيذ الدالة.

Var1_in تمثل المتغير (أو اكثر) الداخل الى جسم الدالة.

Function-name يمثل اسم الدالة وتنطبق عليه نفس شروط الواجب توفرها في اسم المتغير.

نستطيع فتح ملف للبرنامج الفرعي من خلال اختيار file→new→function سيقوم بفتح ملف تحرير نصي لكتابة الدالة بالشكل التالي...

```
function [ output_args ] = Untitled3( input_args )
%UNTITLED3 Summary of this function goes here
%   Detailed explanation goes here

end
```

والمثال التالي يوضح كيفية كتابة دالة بسيطة لجمع رقمين وحفظها باسم الدالة (يجب ان يكون اسم الدالة بعد علامة المساواة مطابق لاسم الدالة عند حفظ الملف على الحاسوب)، ثم استدعاءها من خلال نافذة تنفيذ الاوامر

function [sumxy] = waleed (x,y) **sumxy=x+y** **end**	**>> n=3;** **>> m=5;** **>> waleed(n,m)** **sumxy =** **8**

كما يمكن استدعاء الدالة من خلال ملف ماتلاب مستقل تم كتابته بواسطة M-file ، وكما موضح في المثال التالي حيث سيتم بناء برنامج فرعي لتربيع عناصر مصفوفة احادية تحتوي 5 عناصر ..

File→new→function	**File→new→script**
function [squre1] = squarew(x) **x=x.^2** **end**	n=[1 2 3 4 5]; squarew(n);

حيث سيكون ناتج تنفيذ البرنامج السابق هو

```
x =
     1     4     9    16    25
```

وفي المثال التالي توضيح لكتابة برنامج فرعي يقوم بحساب مجموع عناصر مصفوفة ثنائية وتنفيذ الدالة من نافذة الاوامر.

```
function [ result ] = sum2_d( b )
 sum (sum(b))
end
```

```
>> a=randn(3);
>> sum2_d(a)
ans =
    3.4734
```

ان البرامج الفرعية في ماتلاب تدعم الحصول على اكثر من متغير كمخرجات بعكس العديد من اللغات البرمجية الاخرى، وكما موضح في المثال التالي..

```
function [ r1 , r2 , r3 ] = multi2d( b )
r1 = sum(sum(b));
r2 = max(max(b));
r3 = min(min(b));
end
```

```
>> [ r1 , r2 , r3 ] = multi2d( a )
r1 =
    3.4734
r2 =
    3.5784
r3 =
   -2.2588
```