```
    0.3090    0.5878    0.8090    0.9511    1.0000    0.9511    0.8090
  Columns 8 through 10
    0.5878    0.3090    0.0000
```

ويمكن تفسير الدوارة 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.

مثال: توليد 10 أعداد عشوائية قيمتها (1..10).

```
>> array = randperm (10)
array =
        8    2    10    7    4    3    6    9    5    1
>> for n = array
      x (n) = sin (n * pi / 10);
    end;
>> x
x =
   Columns 1 through 7
      0.3090    0.5878    0.8090    0.9511    1.0000    0.9511    0.8090
   Columns 8 through 10
      0.5878    0.3090    0.0000
```

سيأخذ متغير الحلقة n هنا قيماً عشوائية بين (1) و (10) معطاة بالمصفوفة array.

ملاحظة:

يمكن إنشاء عدة حلقات for متداخلة، كما في المثال التالي:

```
>> for n =1: 5
      for m = 5: -1: 1
        A (n, m) = n ^ 2 + m ^ 2;
      end;
      disp (n);
```

```
end;
```

الإخراج

```
1
2
3
4
5
```

```
>> A
A =
     2    5    10    17    26
     5    8    13    20    29
     10   13   18    25    34
     17   20   25    32    41
     26   29   34    41    51
```

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 (١):

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

x

الإخراج

1

2

3

.

.

24

مثال (٢):

```
>> num = 0; EPS = 1;
>> while (1 + EPS) > 1
     EPS = EPS / 2;
     num = num + 1;
```

```
    end;
>> num
num =
      53
```

ملاحظة:

هناك طريقة قانونية للخروج من حلقة for وwhile وكالاتي:

(في حال تحقق الشرط يتم الخروج من الدوارة for وكذلك while)

```
s = 0;
for i = 1: 100
  s = s + i;
  if s > 250
    break;
 end;
end;
```

الإخراج

```
i = 22
s = 253
```

```
s = 0;
 x = 1;
while x < 100
   s = s + x;
   if  s > 250
     break;
    end;
  x = x + 5;
end;
```

الإخراج

```
x = 51
s = 286
```

ملاحظة:

أذا وجدت الايعاز break ضمن حلقة داخلية واقعة ضمن حلقات اكبر فان البرنامج يخرج من الحلقة التي صادف فيها الايعاز ولا يخرج من الحلقات الأكبر.

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
```

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

إيعاز البت

إضافة إلى المعاملات المنطقية التي ذكرناها سابقاً، يؤمن البرامج توابعاً تسمح بإجراء العمليات المنطقية على بتات منفصلة من الأعداد الصحيحة.

```
>> bitand (3, 4)
ans =
     0
>> bitor (3, 4)
ans =
     7
>> bitxor (13, 27)
ans =
     22
>> bitcmp (20, 5)          متمم العدد ٢٠ لخمس بتات
ans =
    11
 >> bitset (30, 1)          جعل البت الأولى من ٣٠ يكون ١
```