آموزش متلب مقدماتی

مدرس: شهریار سلمانی

تعریف ماتریس

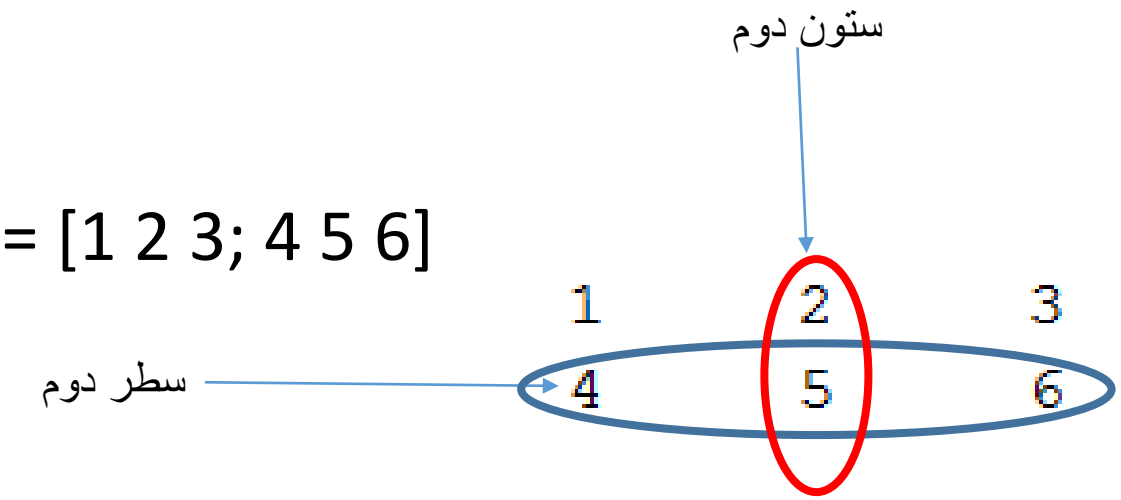
- A=[1 2 3] ← برداری به اندازه 3
- B=[1 2 3; 4 5 6] ← ماتریسی به اندازه 2 سطر و 3 ستون
- C=[1 2 3 4; 5 6 7 8; 9 10 11 12] ← ماتریسی به اندازه 3 سطر و 4 ستون
- D=[1,2,3,4;5,6,7,8] ← ماتریسی به اندازه 2 سطر و 4 ستون

- ویرگول (,) و فاصله (space) هر دو برای جدا کردن درایه‌های ماتریس استفاده می‌شوند.
- از علامت نقطه ویرگول یا semi colon (;) برای رفتن به سطر بعد استفاده می‌شود.

برای مشخص کردن درایه بعدی

برای رفتن به سطر بعد

اندیس‌های ماتریس



- A = [1 2 3; 4 5 6]
- A ماتریسی با ۲ سطر و ۳ ستون است.
- نحوه فراخوانی اعضای ماتریس به ۲ صورت است:
- A(ستون , سطر) → Example: A(2,2) → 5
- A(اندیس مورد نظر) → Example: A(5) → 3
- در فراخوانی اعضا با یک عدد باید توجه داشت، شماره اندیس‌ها به اینصورت جلو می‌روند که ابتدا کل یک سطر را می‌شماریم و سپس سراغ سطر بعدی می‌رویم.

چند نکته

- اگر در آخر دستوراتمان semi colon بگذاریم، نتیجه دستور در Command Window نشان داده نمی‌شود، ولی اگر دستورمان متغیری ساخته باشید، متغیر در Workspace ساخته می‌شود.

```
>> A=[1 2 3; 4 5 6];
>>
```

```
>> A=[1 2 3; 4 5 6]

A =

     1     2     3
     4     5     6

>>
```

- در هر دو مثال روبه‌رو متغیر A در Workspace ساخته می‌شود.

- دستور magic(row, column) برای ما ماتریسی می‌سازد با سایز row در column، که ویژگی این ماتریس این می‌باشد که، جمع هر سطر و ستون برابر است.

```
>> A=1:5

A =

     1     2     3     4     5
```

- علامت دو نقطه(:) به معنی تا است.

چند تابع در متلب

- sum, max, min
- دستورهای بالا، به ترتیب جمع، بیشترین و کمترین عضوها را به صورت ستونی می‌دهند، به این معنی که ابتدا حاصل جمع ستون اول را میدهد، سپس حاصل جمع ستون دوم و به همین ترتیب تا پایان. برای دستورات max و min، بیشترین و کمترین عضو را به صورت ستونی می‌دهند.
- Example:

1

```
>> A=magic(5)

A =

    17    24     1     8    15
    23     5     7    14    16
     4     6    13    20    22
    10    12    19    21     3
    11    18    25     2     9

>> min(A)

ans =

     4     5     1     2     3
```

2

```
>> A=[8 3 6; 3 5 9]

A =

     8     3     6
     3     5     9

>> sum(A)

ans =

    11     8    15
```

به‌کارگیری توابع

- برای به کار بردن توابع در متلب باید توجه داشت که ورودی تابع را درست وارد کنیم.
- برای دیدن نحوه صحیح استفاده از یک تابع در متلب (syntax) از دستور doc استفاده می‌کنیم.

- Example: doc sum
- Example:

```
>> A=magic(5)

A =

    17    24     1     8    15
    23     5     7    14    16
     4     6    13    20    22
    10    12    19    21     3
    11    18    25     2     9

>> min(A)

ans =

     4     5     1     2     3
```

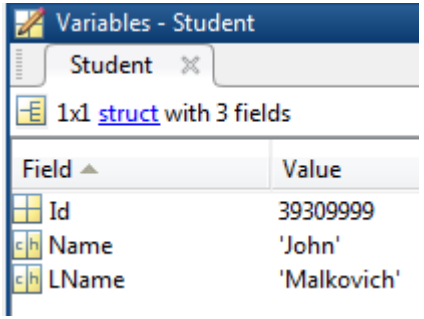
آرایه سلولی (Cell Array)

- مثل ماتریس‌ها است، ولی در اینجا نیازی نیست که ورودی‌های ما همگی از یک جنس باشند.
- Example: A={1 2 3 'M'; 2 6 'Ali' 'Alavi'};

آرایه ساختاری (Structure Array)

- برای اینکه بتوان یک داده منسجم برای یک نوع خاص داشت از این نوع داده استفاده می‌شود.
- در استفاده از این نوع داده، می‌توان برای یک نوع خاص، Field یا زمینه‌های مختلف تعریف کرد.

- Example:
  - Student.Id = 39309999
  - Student.Name = 'John‘
  - Student.Lname = 'Malkovich‘




```
>> Student

Student =

  struct with fields:

      Id: 39309999
    Name: 'John'
   LName: 'Malkovich'
```

دستور شرطی if

- برای بررسی شرط از دستور if استفاده می‌کنیم.
- در دستور if می‌توان برای بررسی حالات دیگر شرط از else و یا elseif استفاده نمود.
- در پایان بررسی شروط از end استفاده می‌کنیم.
- اگر شرط درست باشد، دستورات بین if و end اجرا می‌شوند.

```
if expression
    statements
elseif expression
    statements
else
    statements
end
```

```
>> a=5;
>> if a == 5
b=a*2
elseif a<5
b=a*3
else
b=a*4
end

b =

    10
```

نکته بسیار مهم: علامت مساوی (=) برای اختصاص دادن است،
ولی علامت مساوی مساوی (==) برای مقایسه کردن است.

منطق گذاره‌ها

- True → 1
- False → 0

AND

x y	F= x.y
0 0	0
0 1	0
1 0	0
1 1	1

&&

OR

x y	F= x+y
0 0	0
0 1	1
1 0	1
1 1	1

||

منطق گذاره‌ها

- Example: (AND)

```
>> a=5;
>> b=8;
>> if a<8 && b<10
c=a*b
end

c =

    40
```

```
>> a=5;
>> b=8;
>> if a<8 && b<4
c=a*b
end
```

- Example: (OR)

```
>> a=5;
>> b=8;
>> if a<8 || b<4
c=a*b
end

c =

    40
```

```
>> a=5;
>> b=8;
>> if a<4 || b<7
c=a*b
end
```

عملگرها

- ترتیب عملگرها: پرانتز، توان، ضرب، تقسیم، جمع، تفریق

```
>> A=2*(3+7)/5+3

A =

     7
```

- ضرب (*). تقسیم (/). جمع (+). تفریق (-).

- عملگر توان نیز با استفاده از (^) ممکن می‌شود.

```
>> A=2*(3+7)^2/5+3

A =

    43
```

اعداد تصادفی

- با استفاده از دستور rand می‌توان عدد تصادفی بین صفر و یک تولید کرد.
- Example:

```
>> rand(1,3)

ans =

    0.5285    0.1656    0.6020
```

- که در اینجا عدد اول تعداد سطر و عدد دوم تعداد ستون را مشخص می‌نماید.

در مثال بالا می‌بینیم که rand(1,3) یک بردار با اندازه ۱ سطر و ۳ ستون درست کرده، که مقادیر آن همگی اعداد تصادفی بین صفر و یک هستند.

- با استفاده از دستور randi می‌توان عدد تصادفی صحیح بین یک و یک مقدار ماکزیمم به دست آورد.

اعداد تصادفی

- Examples:

```
>> randi(100)

ans =

    27
```

- عددی تصادفی بین ۱ و ۱۰۰:

```
>> randi([100 200])

ans =

   166
```

- عددی تصادفی بین ۱۰۰ و ۲۰۰:

- ماتریسی با ابعاد ۳ سطر و ۴ ستون، که اعضای آن اعدادی تصادفی صحیح بین ۲۰۰ و ۵۰۰ است.

```
>> randi([200 500], 3, 4)

ans =

   407   225   245   499
   425   268   448   223
   335   474   362   333
```

دستورات تکرار (Loops)

- حلقه تکرار for: برای انجام کارهای تکراری با تعداد معلوم استفاده می‌شود.
- حلقه تکرار while: برای انجام کارهای تکراری با تعداد نامعلوم استفاده می‌شود.
- نکته: دو قید، تکرار معلوم و نامعلوم، قدر مسلم نیستند.

```
for index = values
    statements
end
```

```
while expression
    statements
end
```

دستورات تکرار (Loops)

- Example (For Loop):

```
>> for i=1:10
a(i) = i*2;
end
>> a

a =

     2     4     6     8    10    12    14    16    18    20
```

- Example (While Loop):

```
>> i=1;
>> while i<10
student(i).Id = 39300000 + i*5;
i=i+1;
end
```

Variables - student

student

1x9 struct with 1 field

Fields	Id
1	39300005
2	39300010
3	39300015
4	39300020
5	39300025
6	39300030
7	39300035
8	39300040
9	39300045

استفاده از m-file

- می‌توان با زدن کلید ctrl + N و یا نوشتن edit در محیط command window و یا رفتن به tab اول(Home) و سپس انتخاب New Script → Script به این محیط دسترسی یافت.
- در این محیط می‌توان کدهای مورد نظرمان را بنویسیم و در نهایت به صورت کلی آن را اجرا کنیم، برخلاف Command Window که هر دستور در همان‌جا اجرا می‌شد.

- نکته: در این محیط با استفاده از % در اول خط، می‌توان آن خط را Comment یا توضیحات در نظر گرفت. Comment ها در اجرای برنامه هیچ تاثیری ندارند، ولی بعدها در فهم برنامه به شما کمک می‌کنند.

چند مثال با حلقه‌های تکرار

- تولید دنباله اعداد به شکل زیر:

- 1,0,3,0,5,0,7,0,9,…,19

```
1     % Here We Write Our Comment
2     i=1;
3     while i<20
4
5         if mod(i,2) == 0
6             a(i) = 0;
7         else
8             a(i) = i;
9         end
10
11        i=i+1;
12    end
```

دستور mod برای ما باقیمانده تقسیم را به دست می‌آورد. پس در اینجا وقتی می‌گوییم اگر باقیمانده تقسیم بر دو یک عدد برابر با صفر باشد، یعنی داریم زوج بودن آن را بررسی می‌کنیم.

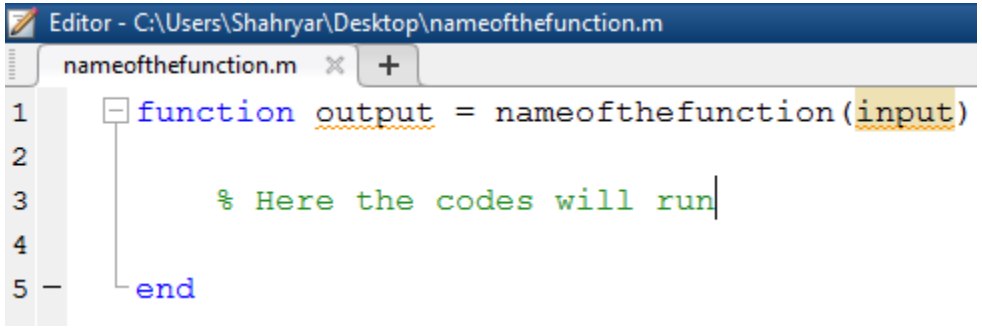
مثال دستور mod:

```
>> mod(100,4)

ans =

     0
```

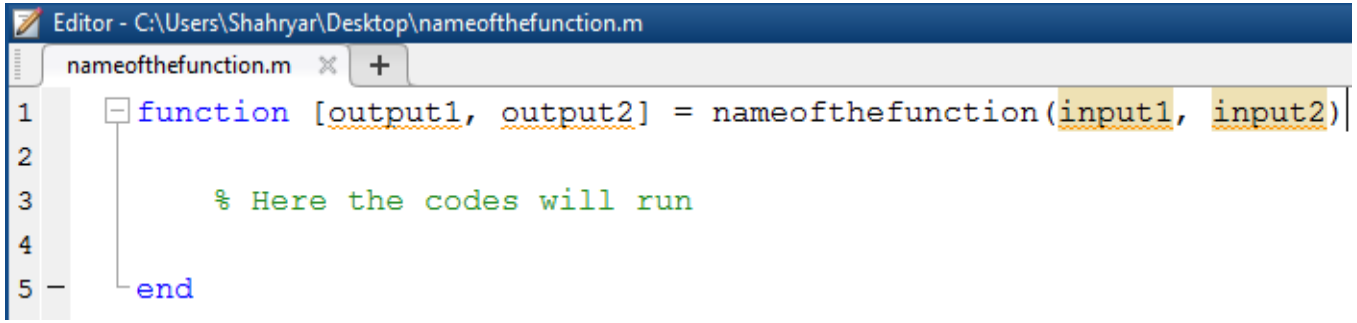
```
>> mod(100,3)

ans =

     1

>> 3 * 33 + 1

ans =

   100
```

نوشتن تابع

- برای نوشتن تابع در متلب باید از قوانین خاصی پیروی کرد. فرمت آن به شکل زیر است.
- نکته: نام تابع با نام فایل ذخیره شده باید هماهنگ باشد.
- تعریف تابع با یک ورودی و یک خروجی



- تعریف تابع با چند ورودی و چند خروجی



مثالی از یک تابع ساده

حل معادله درجه دوم به فرم:

a*x^2 + b*x + c = 0

quaraticequation.m

```
1   function [x1, x2] = quaraticequation(a,b,c)
2
3       % This function will solve the quadratic equation
4       % in form of a*x^2 + b*x + c = 0
5
6       delta = b^2 - 4*a*c;
7
8       x1 = (-b + sqrt(delta)) / (2*a);
9       x2 = (-b - sqrt(delta)) / (2*a);
10
11  end
```