آموزش برنامه نویسی متلب

به نام خدا

آموزش برنامه نویسی متلب

تقدیم به هموطنان عزیزم

متلب را با لذت یاد بگیرا!

آموزش برنامه نویسی متلب

آموزش برنامه نویسی متلب

موضوع: آرایه سلول ها در متلب

جلسه: دهم

مدرس : مدرسین جاواپرو

متلب را ساده،آسان و شیرین بنوشید!!!

آموزش برنامه نویسی متلب

ارتباط با ما:

سایت: www.javapro.ir

ایمیل:RAHMAN.ZARIE92@GMAIL.COM

کانال تلگرام:

@javalike

گروه پرسش و پاسخ برنامه نویسی :

@javapro_ir

آموزش برنامه نویسی متلب

در این جلسه می خواهیم با دو مورد دیگر از انواع داده ها در متلب یعنی آرایه سلول-ها،cells، و ساختارها،structures، آشناشویم. ابتدا با آرایه سلول ها آغاز می کنیم.

آرایه سلول ها همان طور که از نام آن برمی آید یک آرایه متلب از داده هایی به نام سلول می باشد. یک سلول خود حاوی اطلاعات مختلفی ازجمله متغیرها، آرایه ها و حتی سلول-های دیگر باشد. برای مثال یک آرایه سلول می تواند شامل یک ماتریس ۳در۳، یک ماتریس رشته، یک بردار و یک آرایه سلول به شکل زیر باشد:

cell(1,1) [1 2 3; 4 5 6 7 8 9]	cell(1,2) ['Ali'; 'Sajedi'; '987654']
cell(2,1) [1 2 3 4 5 6 7 8]	cell(2,2) … … … …

دیاگرام فوق برای درک بهتر عملکرد آرایه سلولی و همچنین چگونگی اندیس های یک آرایه سلول کمک می کند. این داده برای ذخیره داده های یک موجود با انواع مختلف داده مثلا مشخصات یک دانشجو مناسب می باشد. آرایه سلول ها مانند ماتریس ها با هر اندازه مستطیلی دلخواه قابل تعریف می باشند. البته می توان ماتریس یا آرایه سلول ها را با ابعاد بیشتر نیز مثلا آرایه های ۳ بعدی تعریف نمود.

جهت تعریف یک آرایه سلول به یکی از دو طریق زیر می توان اقدام کرد:

- با استفاده از عملگر تخصیص (=)
- پیش اختصاص دهی با تابع cell و سپس تخصیص داده ها

آموزش برنامه نویسی متلب

-روش اول:

آرایه سلولی با نام A را درنظر بگیرید. می توان با اختصاص داده های دلخواه به تک تک عناصر آرایه، که به طور مناسب اندیس گذاری شده اند آرایه سلول را ایجاد نمود:

```
>>A(1,1) = {[1 2 3;4 5 6;7 8 9]};
>>A(1,2) = {['Ali    ';'Sajedi';'987654']};
>>A(2,1) = {1:8};
>>A(2,2) = { {} };
```

از {} برای ایجاد یک سلول به کار می رود. در هر یک از سلول های آرایه یک نوع داده ذخیره-شده است. هنگامی که خط اول برنامه را وارد کنید، متلب به طور خودکار یک آرایه سلول ادرا ایجاد کرده و داده ها را به آن اختصاص می دهد. سپس در خط دوم با اختصاص داده ها به سلول بعدی، آرایه A به یک آرایه ادر۲ گسترش می یابد. سپس با تکرار این عمل در خط سوم متلب به طور خودکار یک آرایه ۲در۲ را ایجاد می کند و مقدار سلول آخر یعنی چهارم را برابر با یک سلول تهی "{}" قرار می دهد. بنابراین اگر از آخر به اول دستورات فوق را وارد کنید ابتدا متلب یک آرایه ۲در۲ با سلول های تهی ایجاد نموده و سپس مقدار سلول چهارم را اختصاص می دهد. پس گسترش یک آرایه حتی بعد از تعریف آن، ممکن خواهدبود.

روش زیر نیز برای تعریف آرایه فوق معتبر است:

```
>>A{1,1} = [1 2 3;4 5 6;7 8 9];
>>A{1,2} = ['Ali    ';'Sajedi';'987654'];
>>A{2,1} = 1:8;
>>A{2,2} = {};
```

دقت شود که در روش اول اندیس های A داخل پرانتز وارد شده و برای تعریف سلول داده های مربوطه را در داخل {} قرار دادیم اما در روش دوم عکس این عمل انجام شد. هر دو تعریف نتیجه یکسانی خواهند داشت و می توان در هرزمان از روش های فوق به-جای یکدیگر استفاده نمود.

- نکته: یک تفاوت اصلی آرایه سلول با آرایه ها و ماتریس ها در این است که چنان چه یک ماتریس یا آرایه ای که تعریف کرده ایم را دوباره با نام قبلی ولی با مقادیر جدید تعریف کنیم، متلب به طور

خودکار ابتدا داده قبلی را پاک نموده و ماتریس یا آرایه جدید را با ابعاد تازه به متغیر با نام قبلی اختصاص می دهد. اما در مورد سلول ها این گونه نیست. برای مثال، در مثال قبلی با تعریف مجدد A این بار با استفاده از اندیس گذاری با {}، اگر به متغیرهای فضای کاری دقت کنید، آرایه قبلی پاک نمی شود، بلکه تنها سلول های جدید جایگزین سلول های متناظر قبلی می شوند. بنابراین اگر می خواهیم مواردی چون مثال قبل را در یک آرایه جدید وارد کنیم بهتر است با دستور clear ابتدا آرایه را پاک نماییم.

- روش دوم:

با تابع cell(m,n) می توان یک سلول m در n با عناصر سلول تهی ایجاد نموده وسپس به شکل قبل داده ها را به سلول ها تخصیص داد:

```
>>B = cell(2,3)
B =
    []    []    []
    []    []    []
>>B(2,2) = {[1 2;3 4]}
B =
    []              []    []
    []    [2x2 double]    []
```

- دستور whos :

این دستور را برای A به کار برید و نتیجه را تحلیل کنید:

```
>>whos A
  Name        Size              Bytes  Class     Attributes

  A           2x2                 620  cell
```

برای مشاهده محتویات A، نام داده را وارد می کنیم:

```
>>A
```


```
A =
    [3x3 double]    [3x6 char]
    [1x8 double]            {}
```

دیدیم که مانند ماتریس ها محتویات را نمایش نمی دهد و فقط اندازه و نوع داده ها را در چاپ می کند. برای مشاهده محتویات از celldisp() استفاده می شود:

```
>>celldisp(A)
A{1,1} =
     1     2     3
     4     5     6
     7     8     9
A{2,1} =
     1     2     3     4     5     6     7     8
A{1,2} =
Ali
Sajedi
987654
A{2,2} =
     {}
```

برای نمایش یک یا چند تا سلول دلخواه همانند ماتریس ها به دوشکل می توان اندیس گذاری نمود:

۱. به روش شماره سلول:

```
>>A{1,2}
ans =
Ali
```

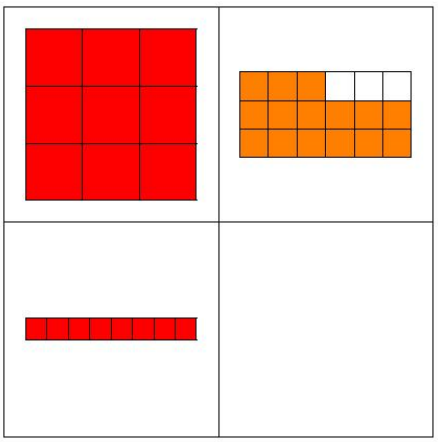

```
Sajedi
987654
```

- دستور cellplot(A) :

این دستور را وارد کرده تا پنجره ای گرافیکی بازشود:

```
>>cellplot(A)
```

پنجره زیر به شکل گرافیکی نمودار محتویات آرایه سلول را نشان می دهد.



- برای تخصیص همزمان به یک سلول آرایه نمی توان از دستوراتی مثل c{:,1} = ... استفاده نمود. اما به جای آن از طریق تعریف سلول با استفاده از پرانتز،c()={...} ،مانند مثال های زیر می توان عمل نمود:

```
>>c(1,[1 3]) = { {[1 2]} , {'Hi'} }
c =
    {1x1 cell}    []    {1x1 cell}
```

دسترسی به زیر آرایه سلول های یک آرایه سلولی مانند زیر امکان پذیر است:

```
>>C = cell(3,3);
>>C(2:3,2:3) = { {[1 2]} {'Hi'}; {} {[1 2;3 4]} };
>>A = C(1:2,1:2)
```


```
>>celldisp(A)
A{1,1} =
     []
A{2,1} =
     []
A{1,2} =
     []
```

در مثال فوق زیر آرایه سلول A که خود یک آرایه سلول می باشد از آرایه سلول C استخراج گردید.

دسترسی به محتویات سلول ها نیز از طریق اندیس گذاری مناسب امکان پذیر است. برای مثال آرایه سلول C در مثال قبل را درنظر بگیرید. فرض کنیم که درایه دوم بردار تعریف شده در سلول C(۲,۲) یا حرف آخر رشته ی ذخیره شده در سلول C(۲,۳) را بخواهیم استخراج کنیم. باترکیب اندیس گذاری های آرایه سلول ها و آرایه ها می توان بدین منظور دست یافت:

```
>>C
C =
    []            []            []
    []    {1x1 cell}    {1x1 cell}
    []            {}    {1x1 cell}
>>C{2,2}{1,1}
ans =
     1     2
>>C{2,2}{1,1}(1,2)
ans =
     2
>>C{2,3}{1,1}(1,end)
ans =
i
```

- حذف سلول های یک آرایه سلول:

با تخصیص ماتریس تهی، [] ، به خانه های سلول، می توان محتویات آن ها را پاک کرد، مثال زیر به پاک کردن سطرها و ستون های اول آرایه C می پردازد:

```
>>C(3,1:3) = {[] [] []}
C =
    []              []    []
    []    {1x1 cell}    {1x1 cell}
    []              []    []
>>C(1:2,3) = {[] []}
C =
    []              []    []
    []    {1x1 cell}    []
    []              []    []
```

- دستور reshape(A,m,n) :

می توان آرایه سلولی را نیز مانند آرایه های دیگر تغییر اندازه داد. به کارگیری این دستور مشابه آرایه ها بوده و باید تعداد سلول ها (یعنی حاصل m*n)پس از تغییر اندازه همچنان ثابت بماند. این دستور فقط برای تغییر اندازه آرایه بوده و نمی توان از آن جهت حذف یا گسترش آرایه استفاده نمود:

```
>>A = cell(3,4);
>>A(1,4) = {'A'};
>>reshape(A,2,6)
ans =
    []    []    []    []      []    []
    []    []    []    []    'A'     []
```

آموزش برنامه نویسی متلب

پیروز و موفق باشید

آموزش برنامه نویسی متلب

سایت آموزشی رایگان جاواپرو

www. JAVAPro.ir

برنامه نویسی را با تجربه شخصی و به زبان خودمونی یاد بگیرید!!!!!

بازدید از کانال

بازدید از سایت

هر روز مفاهیم و مثال های جدید به سایت اضافه می شود برای اطلاع از مطالب جدید روی سایت عضو کانال شوید.