MATLAB SYNTAX TIPS

There are many ways to read data into files here are a few:

Contents

- File input as a matrix
- File input as a table (a mixed-type) matrix:
- Logical indexing

File input as a matrix

```
items = importdata('items.csv')


items =

      data: [22x2 double]
   textdata: {23x3 cell}



items.data


ans =

     9    150
    13     35
   153    200
    50    160
    15     60
    68     45
    27     60
    39     40
    23     30
    52     10
    11     70
    32     30
    24     15
    48     10
    73     40
    42     70
    43     75
    22     80
     7     20
    18     12
```

```
    4     50
   30     10
```

items.textdata

ans =

'item'	'weight'	'value'
'map'	''	''
'compass'	''	''
'water'	''	''
'sandwich'	''	''
'glucose'	''	''
'tin'	''	''
'banana'	''	''
'apple'	''	''
'cheese'	''	''
'beer'	''	''
'suntan cream'	''	''
'camera'	''	''
'T-shirt'	''	''
'trousers'	''	''
'umbrella'	''	''
'waterproof trousers'	''	''
'waterproof overcl...'	''	''
'note-case'	''	''
'sunglasses'	''	''
'towel'	''	''
'socks'	''	''
'book'	''	''

weight = items.data(:,1)

weight =

```
     9
    13
   153
    50
    15
    68
```

```
27
39
23
52
11
32
24
48
73
42
43
22
 7
18
 4
30
```

File input as a table (a mixed-type) matrix:

```
items = readtable('items.csv')

items =

              item           weight    value
    ------------------------   ------    -----

    'map'                        9        150
    'compass'                   13         35
    'water'                    153        200
    'sandwich'                  50        160
    'glucose'                   15         60
    'tin'                       68         45
    'banana'                    27         60
    'apple'                     39         40
    'cheese'                    23         30
    'beer'                      52         10
    'suntan cream'              11         70
    'camera'                    32         30
    'T-shirt'                   24         15
    'trousers'                  48         10
    'umbrella'                  73         40
    'waterproof trousers'       42         70
    'waterproof overclothes'    43         75
    'note-case'                 22         80
```

```
    'sunglasses'                    7          20
    'towel'                        18          12
    'socks'                         4          50
    'book'                         30          10
```

Can be accessed like a normal matrix, but returns a table type

```
items([1 2 4],1)
```

```
ans =

        item
    ----------

    'map'
    'compass'
    'sandwich'
```

```
items([1:5],3)
```

```
ans =

    value
    -----

    150
     35
    200
    160
     60
```

But called in this way they keep the table type so many functions this don't work, ie ('sum(items([1:5],3))')

Instead used the name of the column rather than the index

```
items.value([1:5])
```

```
ans =
```

```
    150
     35
    200
    160
     60
```

```
sum(items.value([1:5]))
```

```
ans =

    605
```

Logical indexing

Selecting items in a matrix can be done either with a positive interger index, or with a boolean string

Using the index

```
items.value([1:5])
```

```
ans =

    150
     35
    200
    160
     60
```

Using a boolean string

```
selection = false(1,length(items.value)); % create a 1 X length vector of falses
selection(1:2:10) = true % make every other index true up to 10 true
```

```
selection =

  Columns 1 through 13

     1     0     1     0     1     0     1     0     1     0     0     0     0
```

```
   Columns 14 through 22

      0      0      0      0      0      0      0      0      0
```

```
items(selection,:)
```

```
ans =

     item        weight     value
   ---------     ------     -----

   'map'            9        150
   'water'        153        200
   'glucose'       15         60
   'banana'        27         60
   'cheese'        23         30
```

```
sum(items.value(selection))
```

```
ans =

   500
```

NOTE: this has to be a 'logical' index – otherwise Matlab wouldn't know whether you meant 1 as in true or 1 as in the first index

So if you have something like this

```
selection = zeros(1,length(items.value)); % create a 1 X length vector of zeros
selection(2:2:10) = 1 % make every other index true up to 10 1
```

```
selection =

  Columns 1 through 13

      0      1      0      1      0      1      0      1      0      1      0      0      0

  Columns 14 through 22

      0      0      0      0      0      0      0      0      0
```

Be sure to put cast it as a logical

```
sum( items.value( logical(selection) ) )

ans =

    290
```