

Carol Shaw (1955)



- Icono da industria dos videoxogos
- Programadora pioneira de videoxogos en Atari (1978)
- Enxeñeira microprocesadora de videoxogos
- Creadora dos videoxogos River Raid (1982, Activision; 1983, Atari 800), Happy Trails (1983, Intellivision)
- Industry Icon Award (2017) polas súas contribucións á industria dos videoxogos

Vectores e matrices de celdas (*cell arrays*)

- Vectores/matrices onde cada elemento pode ser dun tipo distinto.
- Creación: utilizando chaves { }
 - $s=\{'ola',1:3,17,1+3*i\}$; $disp(s\{1\})$: define un vector de 4 celdas
 - Define un vector de 3 celdas:
 - 1) $vc(1)=\{[1\ 2\ 3]\}$ ou $vc\{1\}=[1\ 2\ 3]$
 - 2) $vc(2)=\{'unha cadea'\}$ ou $vc\{2\}='unha cadea'$
 - 3) $vc(3)=\{ones(3)\}$ ou $vc\{3\}=ones(3)$
- Acceso a un elemento: $s\{1\}$ para imprimir: $fprintf('%s', s\{1\})$; $s(1)$ para eliminar: $s(1)=[]$; $s\{1\}=[]$ asigna a matriz baleira

Vectores e matrices de celdas

- Funcións de manipulación:
 - *cell(m,n)*: crea un cell array baleiro de m filas e n columnas
 - *celldisp(ca)*: mostra o contido de tódalas celdas de ca
 - *iscell(ca)*: indica se ca é un vector de celdas
 - *num2cell(v)*: convirte un vector numérico **v** nun cell array
 - *cellstr(s)*: crea un vector de celdas a partires dun vector de caracteres.

Cadeas de caracteres

- Definición: *str='son a cadea 450'*
- Se queremos ter varias cadeas (p.ex. un vector de cadeas) non podemos facer *s=['ola','adeus']* xa que entón *s='olaadeus'*.
- Non podemos facer *s=['ola' ; 'adeus']*, xa que as dúas filas (ou cadeas) teñen que ter a mesma lonxitude
- Por iso o usual é usar unha celda de cadeas: *s={'ola', 'adeus'}*
- Para acceder á 1ª cadea: *disp(s{1})* ou *disp(s(1))*
- O elemento *s{1}* é de tipo *char*, pero *s(1)* é *cell*:
 - *fprintf('%s\n',s{1})* non da erro
 - *fprintf('%s\n',s(1))* si da erro
- Outra opción: *s=char({'ola','adeus'}); s(1,:)->'ola'; s(2,:)->'adeus'*

Funcións de manipulación de cadeas de caracteres (I)

- Función **strsplit**(s,t): divide unha cadea *s* en palabras usando o delimitador *t* (espazo, por defecto). Retorna un vector de celdas, cada elemento é unha cadea de caracteres (palabra)
- O nº de palabras é o nº de elementos do vector de celdas
- Exemplo: `s='valor 3.65'; p=strsplit(s); p{1}='valor'`
`p{2}='3.65';`
- `str2num(p{2})` da 3.65 como número para poder facer operacións. Se `p{2}` non é un número, `str2num(p{2})` da `[ ]` (matriz baleira)
- Outra forma de converter de char a número: `x=str2double(s)`. Se *s* non ten un número, `str2double` da *NaN*
- Función `isnan(x)`: retorna 1 se `x=NaN`, e 0 en caso contrario.

Funcións de manipulación de cadeas de caracteres (II)

- *ischar(str)*: devolve 1 se *str* é unha cadea de caracteres e 0 se non o é.
- *isletter(str)*: devolve un vector de igual dimensión a *str* con 1 se é unha letra do abecedario e 0 en caso contrario.
 - Ex: *s='ola que tal'; isletter(s) -> 1 1 1 0 1 1 1 0 1 1 1*
(Nota: -> denota a resposta que da Matlab á función *isletter*)
- *isdigit(str)*: igual pero con 1 se son díxitos e 0 en caso contrario
- *isspace(str)*: igual que *isletter()* pero con caracteres de espacio.
 - Ex: *s='ola que tal'; isspace(s) -> 0 0 0 1 0 0 0 1 0 0 0*
- *char(x)*, *char(c)*, *char(t1, t2, t3, ...)*: devolve un vector de caracteres a partires de: *x* un vector de enteiros (códigos Unicode), *c* un vector de celdas de caracteres, as cadeas *t1*, *t2*, *t3*
 - Ex: *c={'ola','adeus'}; s=char(c); s(1,:)->ola; s(2,:)->adeus*

Funcións de manipulación de cadeas de caracteres (III)

- *num2str(x,n)*: convirte o número *x* con *n* (opcional) cifras nunha cadea de caracteres.
- *str2num(s)*: convirte a cadea de caracteres *s* a un número (da *[]* se *s* non é un número).
- *str2double(s)*: convirte *s* a número (da *NaN* se *s* non é un número)
- *strcmp('str1','str2')*, *strcmp('str1',C2)*, *strcmp(C1, C2)*: compara as cadeas *str1* e *str2* e devolve 1 se son iguais e senon 0. Se temos o vector de celdas *C1*, compara *str1* con tódolos elementos de *C1*. No último caso compara os dous vectores de celdas. *strcmp* distingue entre maiúsculas e minúsculas; a función *strcmpi* non.
 - Ex: *strcmp('ola','ola')* -> 1; *strcmp('ola',{'ola','adeus'})* -> 1 0;
strcmp({'ola','pepe'},{'ola','adeus'}) -> 1 0
- *strncmp('str1', 'str2', n)*, *strncmp('str', C, n)*, *strncmp(C1, C2, n)*: igual que *strcmp* pero comparando so os *n* primeiros caracteres.

Funcións de manipulación de cadeas de caracteres (IV)

- *strmatch('str', C)* ou *strmatch('str', C, 'exact')*: devolve un vector columna cos índices de *C* onde se atopa *str*.
 - Ex: *C={'N', 'H', 'O', 'He', 'Ca'}*; *strmatch('H', C)* -> vector columna cos índices 2 e 4 (que corresponden ó 'H' e 'He')
 - Ex: *strmatch('H', C, 'exact')*-> vector columna co índice 2
- Lonxitude dunha cadea: *s='ola caracola'*; *numel(s),length(s)* -> 11
- Concatenación de cadeas: *strcat(s1,s2,s3,...)*; *strvcat(s1,s2,s3,...)*
 - Horizontal: *strcat('ola','adeus')* -> 'olaadeus'
 - Vertical: *strvcat('ola','adeus')* ->

ola
adeus
 - Tamén se pode facer concatenando vectores (supoñendo que unha cadea de caracteres é un vector de caracteres).

Funcións de manipulación de cadeas de caracteres (V)

- *strfind(str, patrón)* e *strfind(cellstr, patrón)*: devolve un vector coas posicións de comezo onde se atopou a cadea *patrón* na cadea *str* ou vector de celdas *cellstr*.
 - Ex: *strfind('ola caracola', 'ol')* -> *1 10* (o patrón 'ol' aparece nas posicións 1 e 10 da cadea 'ola caracola')
 - Ex: *strfind({'ola','caracola'}, 'ol')* -> *[1] [6]* (o patrón 'ol' aparece nas posicións 1 e 6 do 1º e 2º elementos do arrai de celdas)
- *regexprep('str', 'expr', 'repstr')*: reemplaza a expresión *expr* por *repstr* na cadea de caracteres *str* e devolve a cadea resultante.
 - Ex: *cad='H2(g)+O2(g)=H2O'; regexprep(cad, '\(g\)', '')* -> *H2+O2=H2O*: substitúe '(g)' pola cadea baleira, é dicir, elimina '(g)'.

Función *textscan*

- Le un arquivo aberto con *fopen* e almacena nun vector de celdas os datos lidos (números ou cadeas de caracteres):
- Sintaxe:
f=fopen('arquivo.dat','r');
c=textscan(f,formato); % le todo o arquivo
c=textscan(f,formato,n); % le n datos
- O formato é como en *fprintf*: %i,%s,%f,%g
- O valor *c* é un vector de celdas: *c{1},...,c{n}*

Función *textscan*: exemplo

```
clear all;
```

```
fid=fopen('taboa.txt');
```

```
t=textscan(fid, '%s', 4); % ler primeira liña
```

```
celldisp(t);
```

```
datos=textscan(fid, '%d %s %s %f'); % ler resto arquivo
```

```
celldisp(datos);
```

```
fclose(fid);
```

posición	símbolo	elemento	pesoAtómico
1	H	HIDRÓXENO	1
6	C	CARBONO	12
7	N	NITRÓXENO	14.01
8	O	OSÍXENO	16
17	Cl	COLORO	35.45

t{1}{1} = posición
t{1}{2} = símbolo
t{1}{3} = elemento
t{1}{4} = pesoAtómico

Olo:

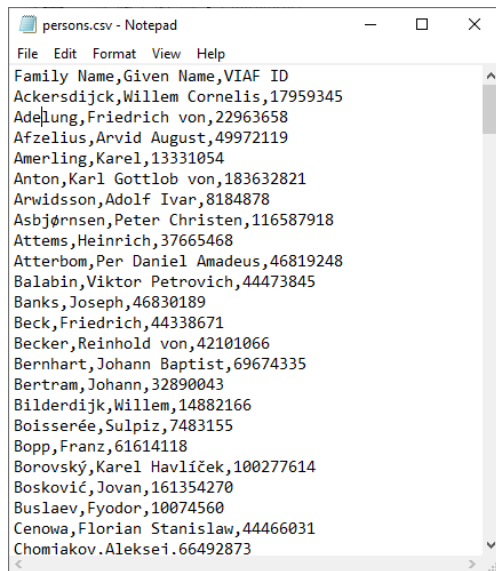
datos{1}(2) -> 6
datos{2}{3} -> 'N'

datos{1}' = [1 6 7 8 17] (vector)
datos{2}' = { 'H' 'C' 'N' 'O' 'Cl'} (vector celdas)
datos{3}' = {'HIDRÓXENO' 'CARBONO' 'NITRÓXENO'
'OSÍXENO' 'COLORO'} (vector celdas)
datos{4}' = [1.0000 12.00 14.010 16.000 35.450] (vector)

Outras funcións de lectura de arquivos

- Arquivos CSV (comma separated values):

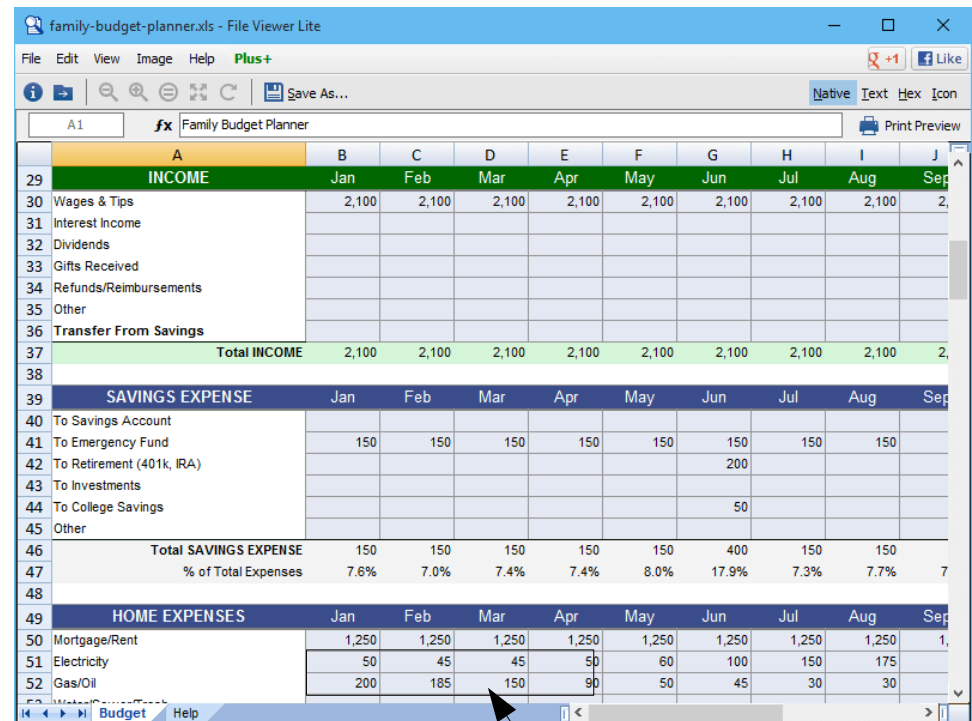
`x=readtable('arquivo.csv')`



Family Name,Given Name,VIAF ID
Ackersdijck,Willem Cornelis,17959345
Adelung,Friedrich von,22963658
Afzelius,Arvid August,49972119
Amerling,Karel,13331054
Anton,Karl Gottlob von,183632821
Arwidsson,Adolf Ivar,8184878
Asbjørnsen,Peter Christen,116587918
Attems,Heinrich,37665468
Atterbom,Per Daniel Amadeus,46819248
Balabin,Viktor Petrovich,44473845
Banks,Joseph,46830189
Beck,Friedrich,44338671
Becker,Reinhold von,42101066
Bernhart,Johann Baptist,69674335
Bertram,Johann,32890043
Bilderdijk,Willem,14882166
Boisserée,Sulpiz,7483155
Bopp,Franz,61614118
Borovský,Karel Havlíček,100277614
Bosković,Jovan,161354270
Buslaev,Fyodor,10074560
Cenowa,Florian Stanislaw,44466031
Chomiakov,Aleksei,66492873

- Arquivos XLS ou XLSX (follas de cálculo):

`x=xlsread('arquivo.xls','páxina','rango')`



	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep
INCOME									
Wages & Tips	2,100	2,100	2,100	2,100	2,100	2,100	2,100	2,100	2,100
Interest Income									
Dividends									
Gifts Received									
Refunds/Reimbursements									
Other									
Transfer From Savings									
Total INCOME	2,100	2,100	2,100	2,100	2,100	2,100	2,100	2,100	2,100
SAVINGS EXPENSE									
To Savings Account									
To Emergency Fund	150	150	150	150	150	150	150	150	150
To Retirement (401k, IRA)						200			
To Investments									
To College Savings						50			
Other									
Total SAVINGS EXPENSE	150	150	150	150	150	400	150	150	150
% of Total Expenses	7.6%	7.0%	7.4%	7.4%	8.0%	17.9%	7.3%	7.7%	7.7%
HOME EXPENSES									
Mortgage/Rent	1,250	1,250	1,250	1,250	1,250	1,250	1,250	1,250	1,250
Electricity	50	45	45	50	60	100	150	175	175
Gas/Oil	200	185	150	90	50	45	30	30	30

Estructuras (*struct*)

- Conxunto de datos heteroxéneo (ten datos de distintos tipos, p.ex. real e carácter)

```
a=struct('nome','Carlos','idade',19);
```

- *a* é unha estrutura con dúas variábeis: *a.nome* é unha cadea, e *a.idade* é un número. Podes imprimir os seus campos con:

```
fprintf('nome=%s idade=%i\n',a.nome,a.idade)
```

- Normalmente se usa un vector ou matriz de estruturas:

```
v=cell(1,10);
```

```
for i=1:10
```

```
    v{i}=struct('nome',nome(i),'idade',idade(i));
```

```
end
```

- Logo podes facer *v{i}.nome* (accedes ao nome *i*-ésimo) ou *[v{:}].nome* (accedes a todos os nomes): *fprintf('%s ',[v{:}].nome)*
- É máis cómodo que usar un vector distinto para cada campo.