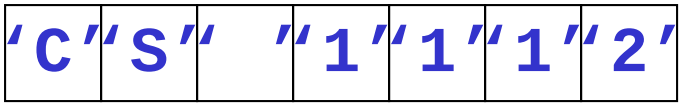
- Previous Lecture:
  - Cell arrays and File I/O
- Today's Lecture:
  - More on topics from previous lectures:
    - Cell arrays
    - File I/O
  - Built-in function sort
- Announcements:
  - Project 5 due Thurs Nov 3rd at 11pm
  - Prelim 2 is two weeks away, on Nov 10th. Email Randy Hess (rbh27) now if you have an exam conflict and state the conflicting event (include the contact info of the instructor in the conflicting course).

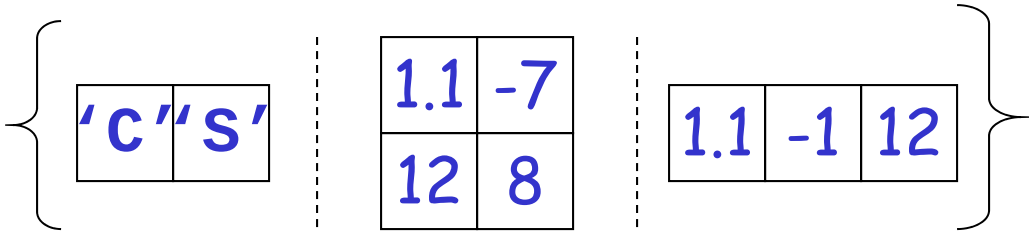
Array vs. Cell Array

- Simple array



  - Each component stores one scalar. E.g., one char, one double, or one uint8 value
  - All components have the same type

- Cell array



  - Each cell can store something "bigger" than one scalar, e.g., a vector, a matrix, a string (vector of chars)
  - The cells may store items of different types

I want to put in the 3rd cell of cell array C a single string. Which is correct?

A. C{3} = 'a cat';

B. C{3} = ['a ' 'cat'];

C. C(3) = {'a ' 'cat'};

D. Two answers above are correct

E. Answers A, B, C are all correct

Example: Build a cell array of Roman numerals for 1 to 3999

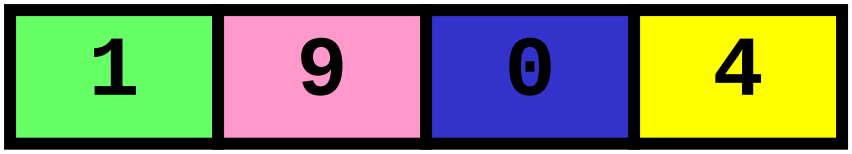
```
C{1} = 'I'
C{2} = 'II'
C{3} = 'III'
   :
C{2007} = 'MMVII'
   :
C{3999} = 'MMMXMXCIX'
```

Example

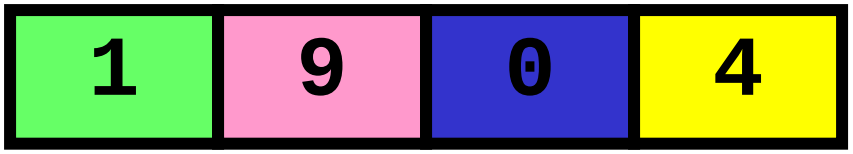
1904 = 1*1000 + 9*100 + 0*10 + 4*1

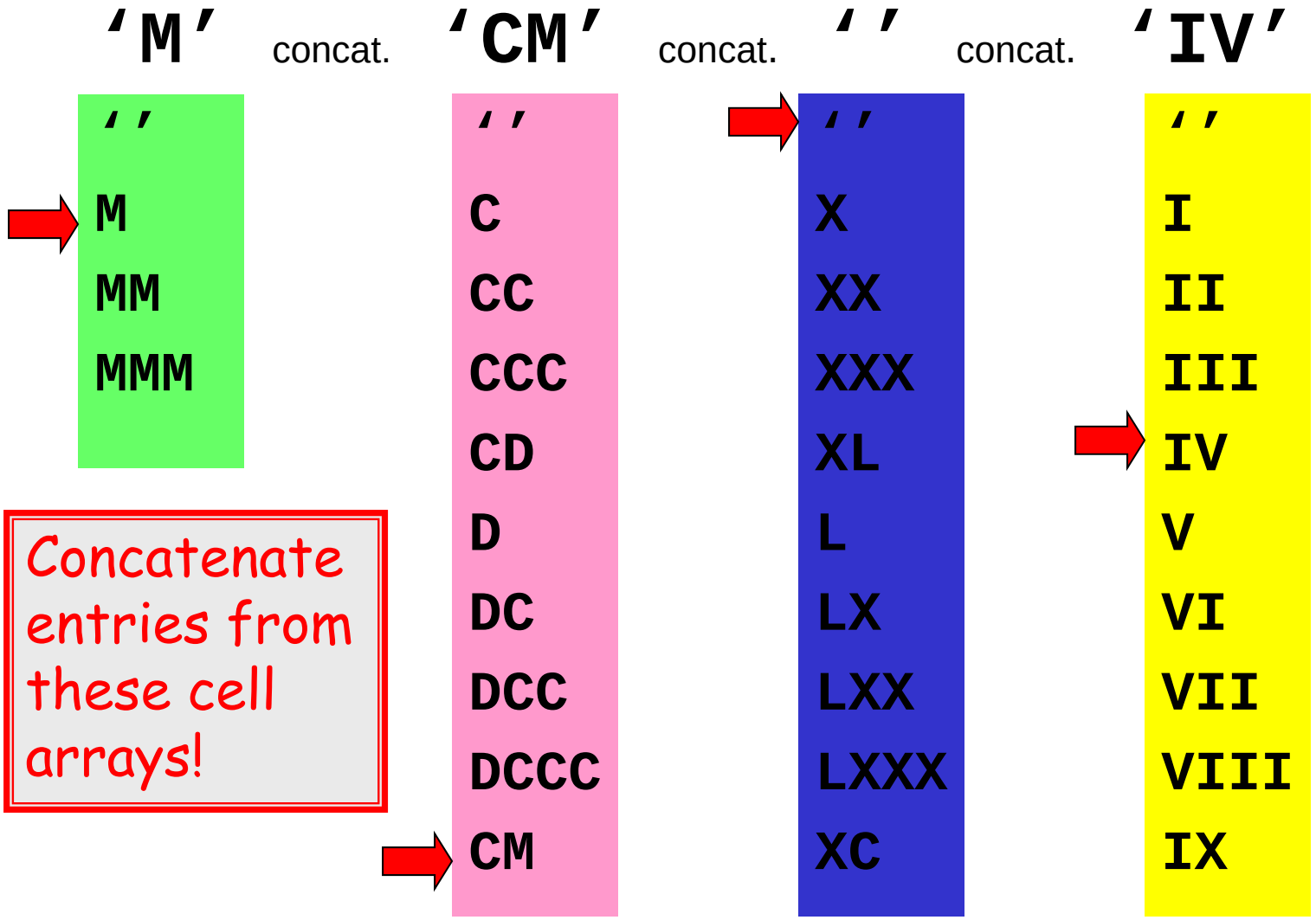
= M CM IV

= MCMIV



MCMIV


1
9
0
4


'M'
concat.
'CM'
concat.
''
concat.
'IV'
''
M
MM
MMM
''
C
CC
CCC
CD
D
DC
DCC
DCCC
CM
''
X
XX
XXX
XL
L
LX
LXX
LXXX
XC
''
I
II
III
IV
V
VI
VII
VIII
IX
Concatenate entries from these cell arrays!

Ones-Place Conversion

```
function r = Ones2R(x)
% x is an integer that satisfies
%       0 <= x <= 9
% r is the Roman numeral with value x.

Ones = {'I', 'II', 'III', 'IV', ...
         'V', 'VI','VII', 'VIII', 'IX'};

if x==0
     r = '';
else
     r = Ones{x};
end
```

Similarly, we can implement these functions:

```
function r = Tens2R(x)
% x is an integer that satisfies
%       0 <= x <= 9
% r is the Roman numeral with value 10*x.
```

```
function r = Hund2R(x)
% x is an integer that satisfies
%       0 <= x <= 9
% r is the Roman numeral with value 100*x
```

```
function r = Thou2R(x)
% x is an integer that satisfies
%       0 <= x <=3
% r is the Roman numeral with value 1000*x
```

We want all the Roman Numerals from 1 to 3999. We have the functions Ones2R, Tens2R, Hund2R, Thou2R.

The code to generate all the Roman Numerals will include loops—nested loops. How many are needed?

A: 2 B: 4 C: 6 D: 8

Now we can build the Roman numeral cell array for 1,…,3999

```
for a = 0:3
  for b = 0:9
    for c = 0:9
      for d = 0:9
        n = a*1000 + b*100 + c*10 + d;
        if n>0
          C{n} = [Thou2R(a) Hund2R(b)...
                                Tens2R(c) Ones2R(d)];
        end
      end
    end
  end
end
```

Now we can build the Roman numeral cell array for 1,…,3999

```
for a = 0:3  % possible values in thous place
  for b = 0:9  % values in hundreds place
    for c = 0:9  % values in tens place
      for d = 0:9  % values in ones place
        n = a*1000 + b*100 + c*10 + d;
        if n>0
          C{n} = [Thou2R(a) Hund2R(b)...
                  Tens2R(c) Ones2R(d)];
        end
      end
    end
  end
end
```

Four strings concatenated together

The n^{th} component of cell array C

Example: subset of clicker IDs

IDs

```
['d091314'; ...
 'h134d83'; ...
 'h4567s2'; ...
 'fr83209']
```

Find subset that begins with ‘h’

L

```
{'h134d83', ...
 'h4567s2'}
```

```
L= {};
k= 0;
for r=1:size(IDs,1)
  if IDs(r,1)=='h'
    k= k+1;
    L{k }= IDs(r,:);
  end
end
```

Directly assign into a particular cell—good!

```
L= {};

for r=1:size(IDs,1)
  if IDs(r,1)=='h'

    L= [L, IDs(r,:)];
  end
end
```

Concatenate cells or cell arrays—prone to problems!

A detailed sort-a-file example

Suppose each line in the file

`statePop.txt`

is structured as follows:

Cols 1-14: State name

Cols 16-24: Population (millions)

The states appear in alphabetical order.

Alabama	4557808
Alaska	663661
Arizona	5939292
Arkansas	2779154
California	36132147
Colorado	4665177
:	:
:	:
Texas	22859968
Utah	2469585
Vermont	623050
Virginia	7567465
Washington	6287759
West Virginia	1816856
Wisconsin	5536201
Wyoming	509294

A detailed sort-a-file example

Create a new file

`statePopSm2Lg.txt`

that is structured the same as **`statePop.txt`**

except that *the states are ordered from smallest to largest according to population.*

`statePop.txt`

```
Alabama          4557808
Alaska            663661
Arizona          5939292
Arkansas         2779154
California      36132147
Colorado         4665177
   :                :
   :                :
```

- Need the pop as *numbers* for sorting.
- Can't just sort the pop—have to maintain association with the state names.

First, read the file and store each line in a cell of a cell array

```
C = file2cellArray('StatePop');
```

```
function CA = file2cellArray(fname)
% fname is a string that names a .txt file
%   in the current directory.
% CA is a cell array with CA{k} being the
%   k-th line in the file.

fid= fopen([fname '.txt'], 'r');
k= 0;
while ~feof(fid)
   k= k+1;
   CA{k}= fgetl(fid);
end
fclose(fid);
```

C

cell array
of strings
in alpha. order

C

```
'Alab   4558000'
'Alas    664000'
   :
'Cali  36132000'



'Verm   623000'
   .
   .
   .
'Wyom   509000'
```

cell array
of strings
in alpha. order

Crew

```
'Wyom   509000'
'Verm   623000'
   .
   .
   .
'Cali  36132000'
```

C

{ 'Alab 4558000'
'Alas 664000'
⋮
'Cali 36132000'
'Verm 623000'
⋮
'Wyom 509000' }

cell array
of strings
in alpha. order

Pop

[4558000
664000
⋮
36132000
⋮
623000
⋮
509000]

vector
of numbers

Crew

{ 'Wyom 509000'
'Verm 623000'
⋮
'Cali 36132000' }

Next, get the populations into a numeric vector

```
C = file2cellArray('StatePop');
n = length(C);
pop = zeros(n,1);
for i=1:n
   S = C{i};
   pop(i) = str2double(S(16:24));
end
```

Converts a string representing a numeric value (digits, decimal point, spaces) to the numeric value → scalar of type double. E.g., `x=str2double(' 3.24  ')` assigns to variable `x` the numeric value 3.24

C

{ 'Alab 4558000'
'Alas 664000'
⋮
'Cali 36132000'
'Verm 623000'
⋮
'Wyom 509000' }

cell array
of strings
in alpha. order

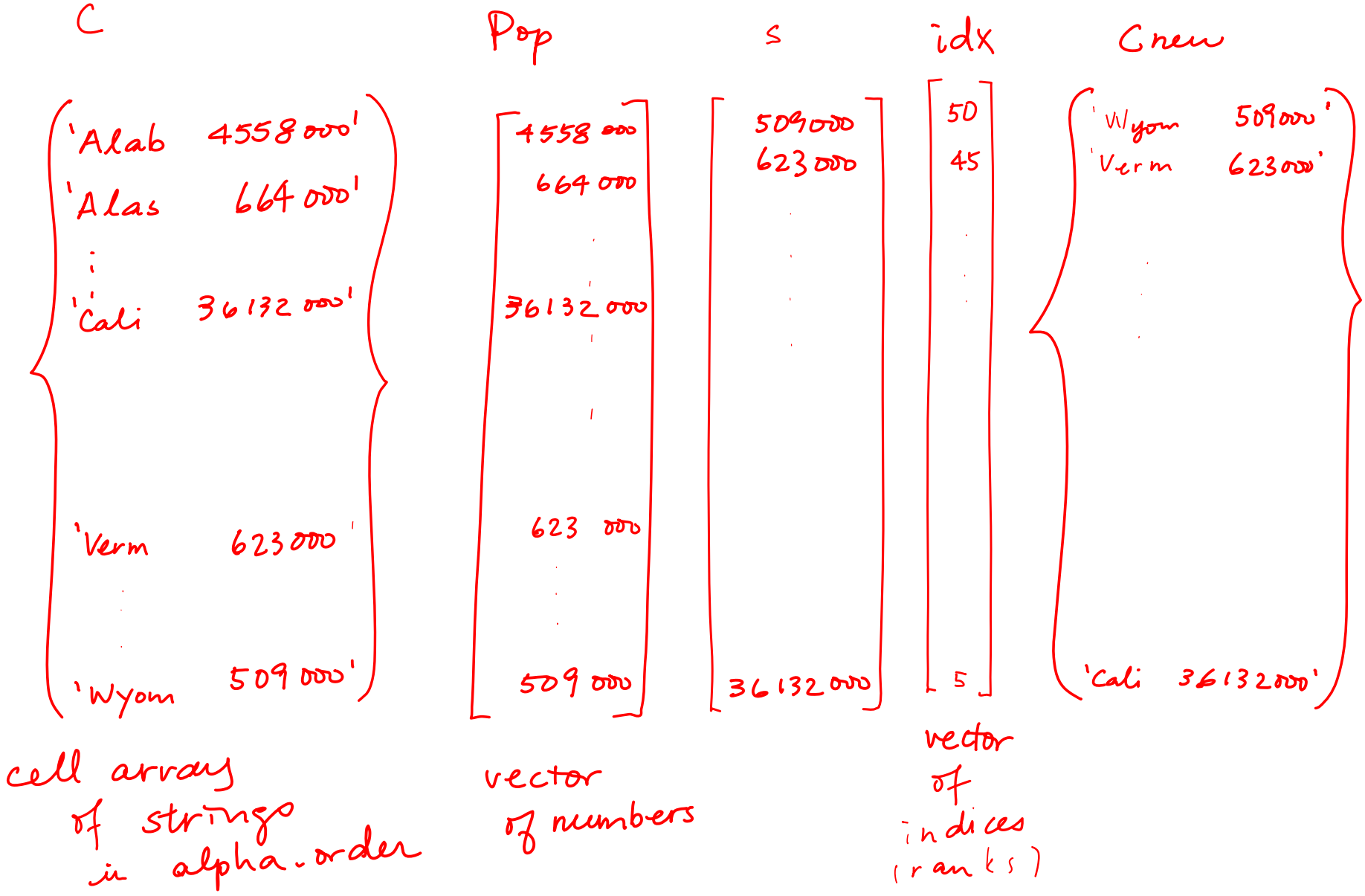
Pop

[4558000
664000
⋮
36132000
⋮
623000
⋮
509000]

vector
of numbers

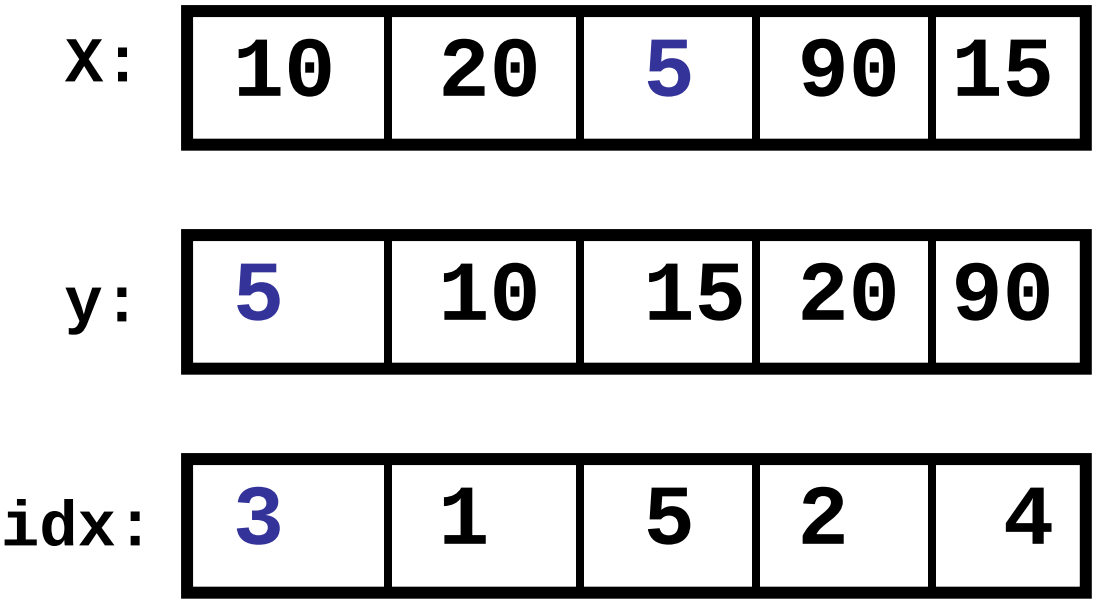
Crew

{ 'Wyom 509000'
'Verm 623000'
⋮
'Cali 36132000' }


C
Pop
s
idx
Cnew
'Alab 4558000'
'Alas 664000'
'Cali 36132000'
'Verm 623000'
'Wyom 509000'
4558000
664000
36132000
623000
509000
509000
623000
36132000
50
45
5
'Wyom 509000'
'Verm 623000'
'Cali 36132000'
cell array of strings in alpha. order
vector of numbers
vector of indices (ranks)

Built-In function `sort`

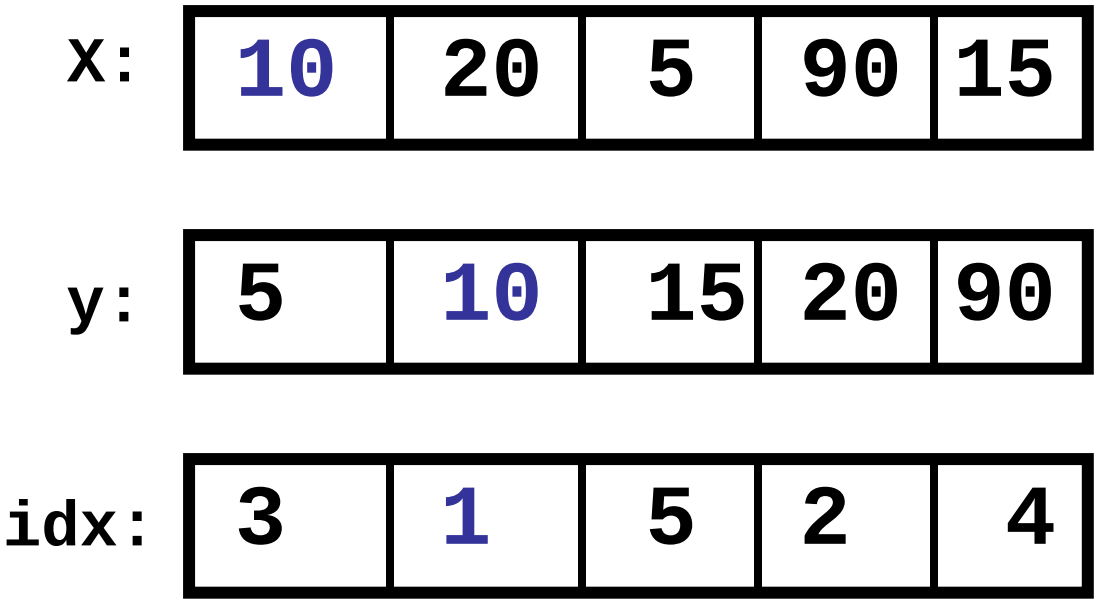
Syntax: `[y,idx] = sort(x)`




```
y(1) = x(3) = x(idx(1))
```

Built-In function `sort`

Syntax: `[y,idx] = sort(x)`



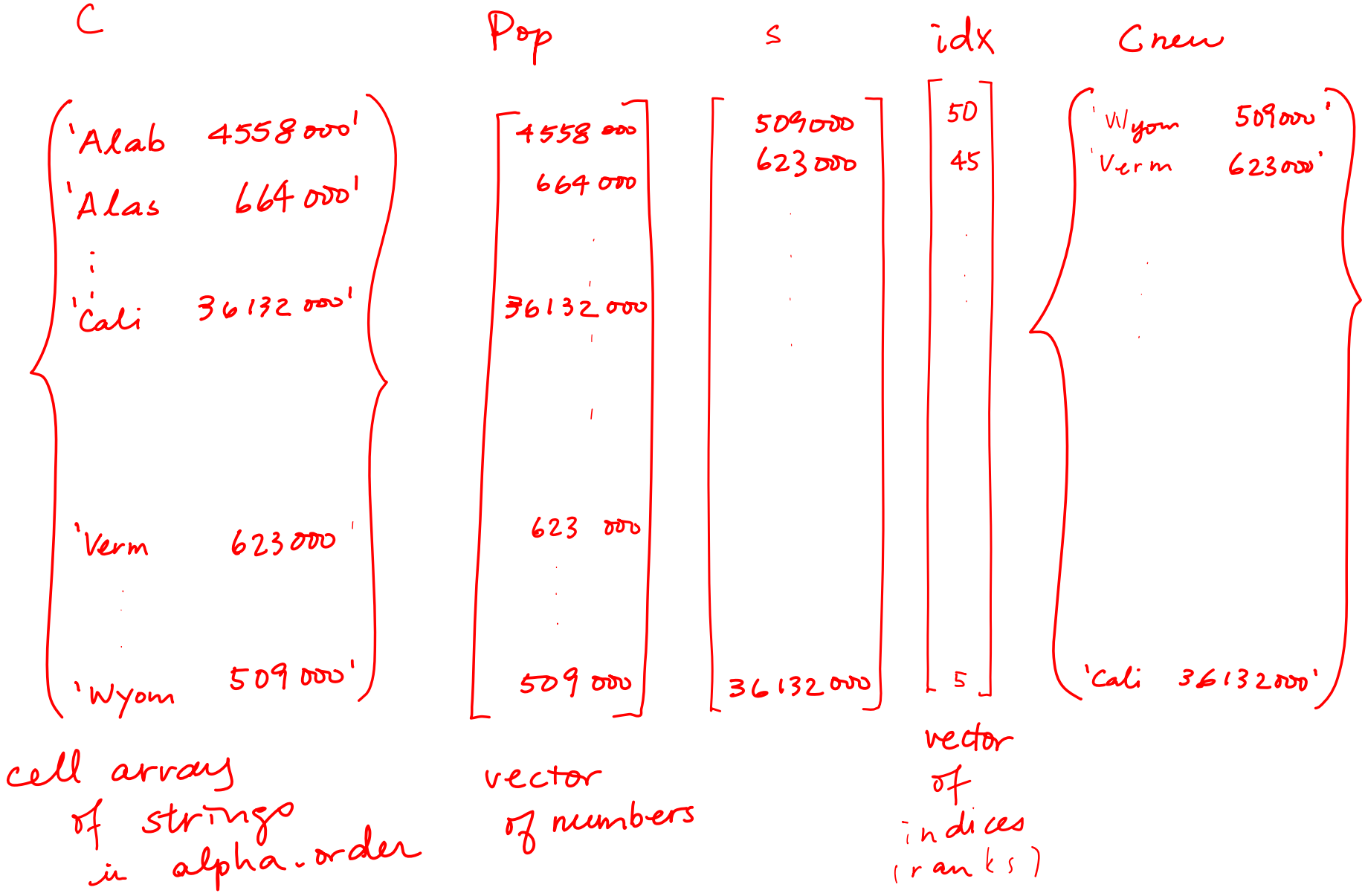

```
y(2) = x(1) = x(idx(2))
```

Built-In function `sort`

Syntax: `[y,idx] = sort(x)`

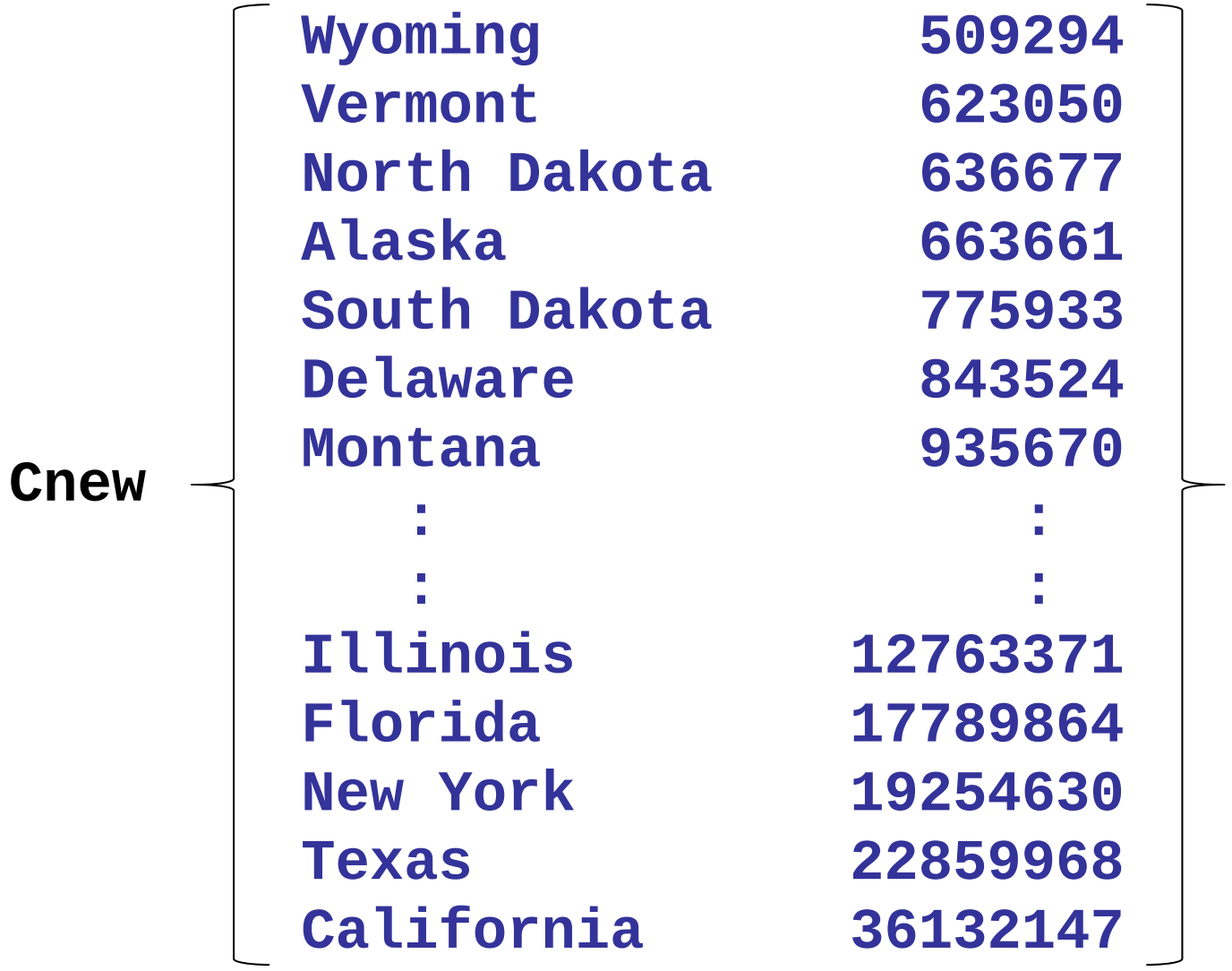
x:	10	20	5	90	15
y:	5	10	15	20	90
idx:	3	1	5	2	4

```
y(k) = x(idx(k))
```


C
Pop
s
idx
Cnew
'Alab 4558000'
'Alas 664000'
'Cali 36132000'
'Verm 623000'
'Wyom 509000'
4558000
664000
36132000
623000
509000
509000
623000
36132000
50
45
5
'Wyom 509000'
'Verm 623000'
'Cali 36132000'
cell array of strings in alpha. order
vector of numbers
vector of indices (ranks)

Sort from little to big

```
% C is cell array read from statePop.txt
% pop is vector of state pop (numbers)
[s,idx] = sort(pop);
Cnew = cell(n,1);
for i=1:length(Cnew)
    ithSmallest = idx(i);
    Cnew{i} = C{ithSmallest};
end
```


Cnew
Wyoming 509294
Vermont 623050
North Dakota 636677
Alaska 663661
South Dakota 775933
Delaware 843524
Montana 935670
: :
: :
Illinois 12763371
Florida 17789864
New York 19254630
Texas 22859968
California 36132147

Sort from little to big

```
% C is cell array read from statePop.txt
% pop is vector of state pop (numbers)
[s,idx] = sort(pop);
Cnew = cell(n,1);
for i=1:length(Cnew)
    ithSmallest = idx(i);
    Cnew{i} = C{ithSmallest};
end
```

```
cellArray2file(Cnew,'statePopSm2Lg')
```

A 3-step process to read data from a file or write data to a file

1. (Create and) open a file
2. Read data from or write data to the file
3. Close the file

1. Open a file

```
fid = fopen('popSm2Lg.txt', 'w');
```

An open file has a file ID, here stored in variable **fid**

Built-in function to open a file

Name of the file (created and) opened. **txt** and **dat** are common file name extensions for plain text files

'**w**' indicates that the file is to be opened for **w**riting

Use '**a**' for **a**ppending

2. Write (print) to the file

```
fid = fopen('popSm2Lg.txt', 'w');

for i=1:length(Z)
    fprintf(fid, '%s\n', Z{i});
end
```

Printing is to be done to the file with ID **fid**

Substitution sequence specifies the *string* format (followed by a new-line character)

The **i**th item in cell array **Z**

3. Close the file

```
fid = fopen('popSm2Lg.txt' ,'w');

for i=1:length(Z)
    fprintf(fid, '%s\n', Z{i});
end

fclose(fid);
```

```
function cellArray2file(CA, fname)
% CA is a cell array of strings.
% Create a .txt file with the name
% specified by the string fname.
% The i-th line in the file is CA{i}

fid= fopen([fname '.txt'], 'w');
for i= 1:length(CA)
    fprintf(fid, '%s\n', CA{i});
end
fclose(fid);
```