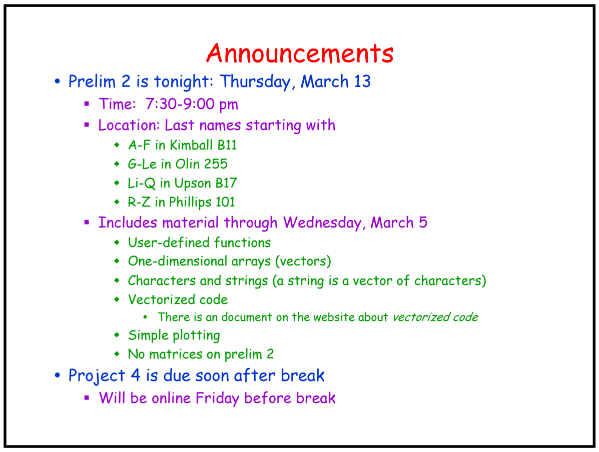
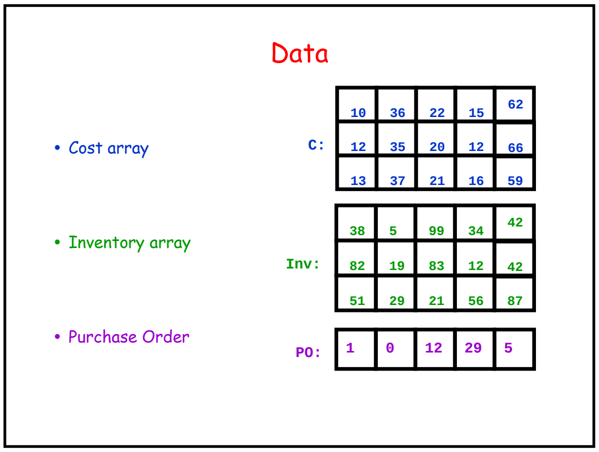


Announcements

- Prelim 2 is tonight: Thursday, March 13
  - Time: 7:30-9:00 pm
  - Location: Last names starting with
    - A-F in Kimball B11
    - G-Le in Olin 255
    - Li-Q in Upson B17
    - R-Z in Phillips 101
  - Includes material through Wednesday, March 5
    - User-defined functions
    - One-dimensional arrays (vectors)
    - Characters and strings (a string is a vector of characters)
    - Vectorized code
      - There is an document on the website about *vectorized code*
    - Simple plotting
    - No matrices on prelim 2
- Project 4 is due soon after break
  - Will be online Friday before break



Data

- Cost array

C:

10	36	22	15	62
12	35	20	12	66
13	37	21	16	59

- Inventory array

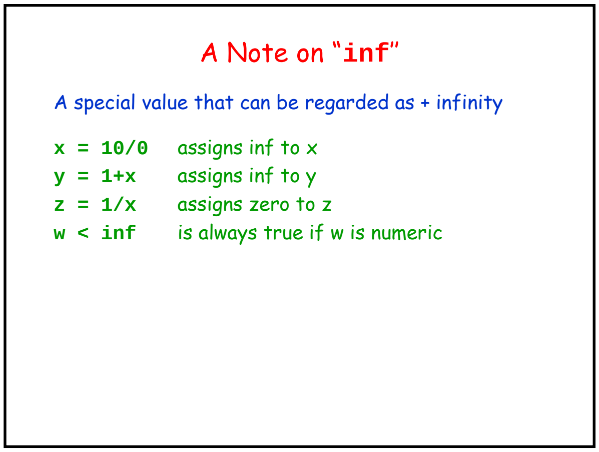
Inv:

38	5	99	34	42
82	19	83	12	42
51	29	21	56	87

- Purchase Order

PO:

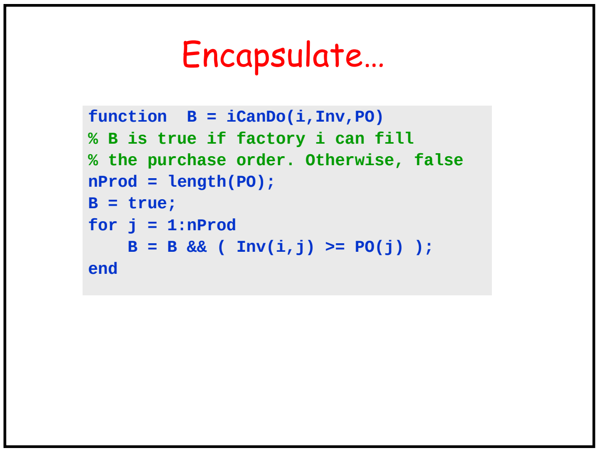
1	0	12	29	5



A Note on "inf"

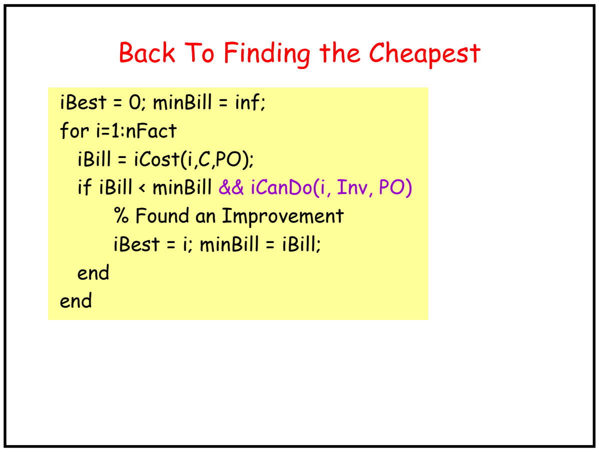
A special value that can be regarded as + infinity

```
x = 10/0    assigns inf to x
y = 1+x     assigns inf to y
z = 1/x     assigns zero to z
w < inf     is always true if w is numeric
```



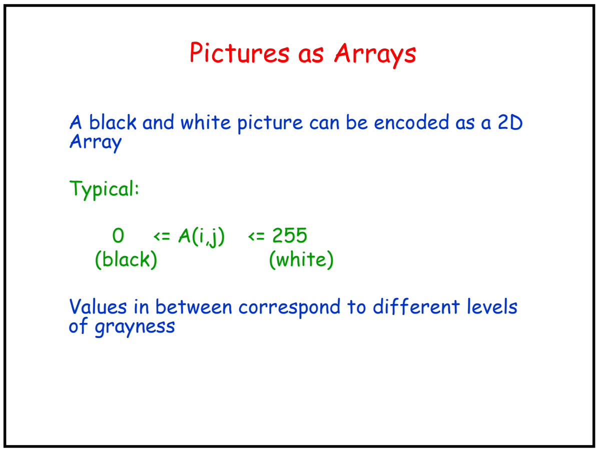
Encapsulate…

```
function  B = iCanDo(i,Inv,PO)
% B is true if factory i can fill
% the purchase order. Otherwise, false
nProd = length(PO);
B = true;
for j = 1:nProd
    B = B && ( Inv(i,j) >= PO(j) );
end
```



Back To Finding the Cheapest

```
iBest = 0; minBill = inf;
for i=1:nFact
  iBill = iCost(i,C,PO);
  if iBill < minBill && iCanDo(i, Inv, PO)
      % Found an Improvement
      iBest = i; minBill = iBill;
  end
end
```



Pictures as Arrays

A black and white picture can be encoded as a 2D Array

Typical:

0 <= A(i,j) <= 255
(black) (white)

Values in between correspond to different levels of grayness

A Color Picture is Represented by 3 Arrays

Stack them in a single 3D array

Typical:

```
0 <= A(i,j,1) <= 255   (red)
0 <= A(i,j,2) <= 255   (green)
0 <= A(i,j,3) <= 255   (blue)
```

↑

Note 3rd Subscript

Solution Framework

1. Read LawSchool.jpg from memory and convert it into an array
2. Manipulate the Array
3. Convert the array to a jpg file and write it to memory

imread

```
% Read image and convert to a 3D array…

>> A = imread('LawSchool.jpg');
>> [m,n,p] = size(A)

m =
      1458      ← rows
n =
      2084      ← columns
p =
         3      ← layers
```

Code for Left-Right Mirror Image

```
for j=1:n
        B(:,j,1) = A(:,n+1-j,1)
        B(:,j,2) = A(:,n+1-j,2)
        B(:,j,3) = A(:,n+1-j,3)
end
```

```
B = A(:,n:-1:1,:);
```

Code for Upside Down Image

```
for j=1:n
        C(i,:,1) = A(m+1-i,:,1)
        C(i,:,2) = A(m+1-i,:,2)
        C(i,:,3) = A(m+1-i,:,3)
end
```

```
C = A(m:-1:1,:,:);
```

Black & White Images and Negatives

- rgb2gray

```
A = imread('LawSchool.jpg');
bwA = rgb2gray(A);
imwrite(bwA,'LawSchoolBW.jpg')
```

- Idea for producing a negative

If matrix A represents the image and

B(i,j) = 255 – A(i,j)

for all i and j, then B will represent the negative