

- Previous Lecture:
 - “Divide and conquer” strategies
 - Binary search
 - Merge sort
- Today’s Lecture:
 - “Divide and conquer” strategies—recursion
 - Merge sort
 - Sierpinski Triangle, revisited
- Announcements
 - Section in classrooms this week
 - CS100M final will be 5/8 (Thurs) 9am. Tell us now if you have a final exam conflict. Email Kelly Patwell with your complete exam schedule (course #s and times)

- CS101J:**
On-line course material; in-person scheduled exams
- Two (or three) options:
- Register and work thru in FALL 08. You can take ENGRD/CS 211 in Spring 09.
 - Learn the material on your own in Summer 08, then in Fall 08 register + submit assignments + take the tests within the first two weeks. You can take CS211 FA08.
 - Possibility: CS offers CS101J in the summer (5/28-6/19). You must be on campus to take the tests during this period.

```
function y = mergeSort(x)
% x is a vector. y is a vector
% consisting of the values in x
% sorted from smallest to largest.

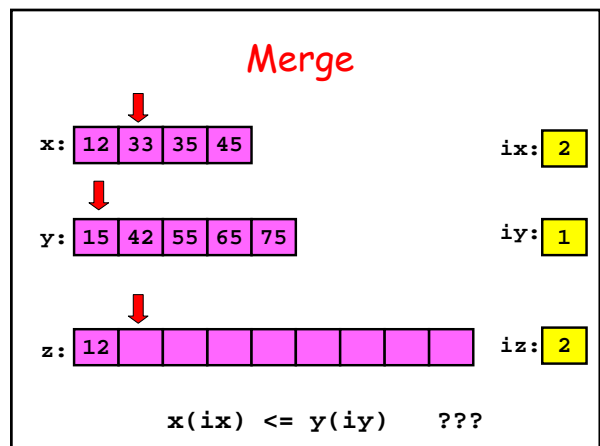
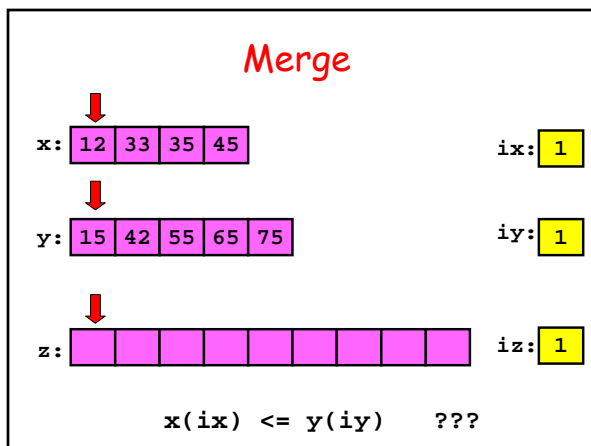
n = length(x);
if n==1
    y = x;
else
    m = floor(n/2);
    y1 = mergeSort(x(1:m));
    y2 = mergeSort(x(m+1:n));
    y = merge(y1,y2);
end
```

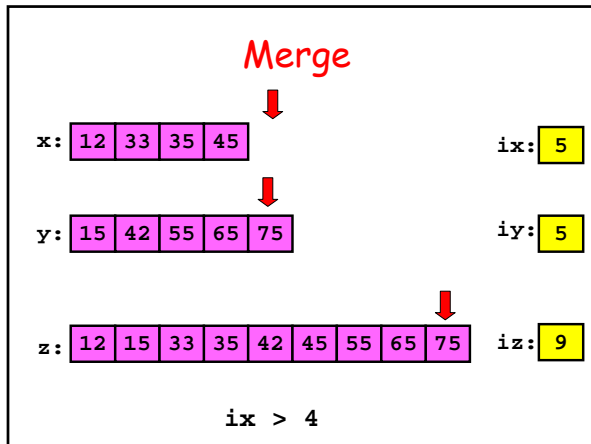
An important sub-problem is the merging of two sorted arrays into one single sorted array

12	33	35	45
----	----	----	----

15	42	55	65	75
----	----	----	----	----

12	15	33	35	42	45	55	65	75
----	----	----	----	----	----	----	----	----





```
function z = merge(x,y)
n = length(x); m = length(y);
z = zeros(1,n+m);
ix = 1; iy = 1;
for iz=1:(n+m)
    if ix > n
        z(iz)=
    elseif iy>m
        z(iz)=
    elseif x(ix) <= y(iy)
        z(iz)=
    else
        z(iz)=
    end
end
```

Why is mesh generation a divide & conquer process?

Let's draw this graphic

April 22, 2008 Lecture 25 43

The basic operation

```
if the triangle is big enough
    Connect the midpoints.
    Color the interior triangle mauve.
else
    Color the whole triangle yellow.
end
```

April 22, 2008 Lecture 25 44

Now, climb your way out.

April 22, 2008 Lecture 25 50

```
function drawTriangle(x,y,level)
% Draw recursively colored triangles.
% x,y are 3-vectors that define the vertices of a triangle.

if level==5
    % Recursion limit (depth) reached
    fill(x,y,'y') % Color whole triangle yellow
else
    % Draw the triangle...
    plot([x x(1)],[y y(1)],'k')
    % Draw and color the interior triangle mauve..
    a = [(x(1)+x(2))/2 (x(2)+x(3))/2 (x(3)+x(1))];
    b = [(y(1)+y(2))/2 (y(2)+y(3))/2 (y(3)+y(1))];
    pause
    fill(a,b,'m')
    pause
    % Apply the process to the three "corner" triangles...
    drawTriangle([x(1) a(1) a(3)], [y(1) b(1) b(3)], level+1)
    drawTriangle([x(2) a(2) a(1)], [y(2) b(2) b(1)], level+1)
    drawTriangle([x(3) a(3) a(2)], [y(3) b(3) b(2)], level+1)
end
```

April 22, 2008 Lecture 25 55