- Previous Lecture:
  - Iteration using **`while`**

- Today’s Lecture:
  - Nested loops
  - Developing algorithms

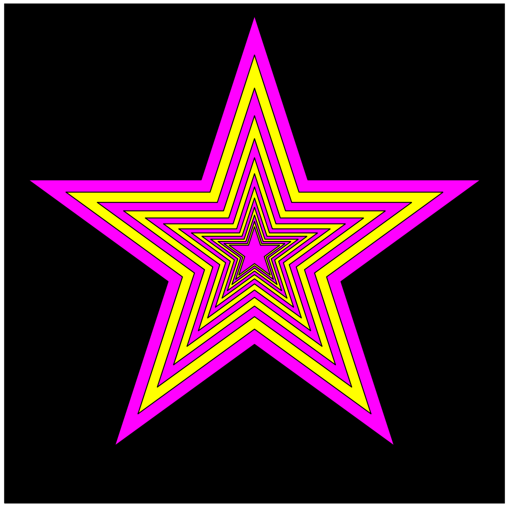
- Announcements:
  - Discussion this week in the lab, UPS B7. Read *Insight* §3.2 beforehand
  - Project 2 due Thurs 2/17 at 11pm
  - We do not use **`break`** in this course
  - Make use of office hrs and consulting hrs
  - Email rbhess@cs.cornell.edu NOW if you have a Prelim 1 conflict

What is the last line of output?

```
x = 1;
disp(x)
y = x;
while y==x && x<=4 && y<=4
   x = 2*x;
   disp(x)
end
```

A: 1 B: 2 C: 4 D: 8

Example: Nested Stars

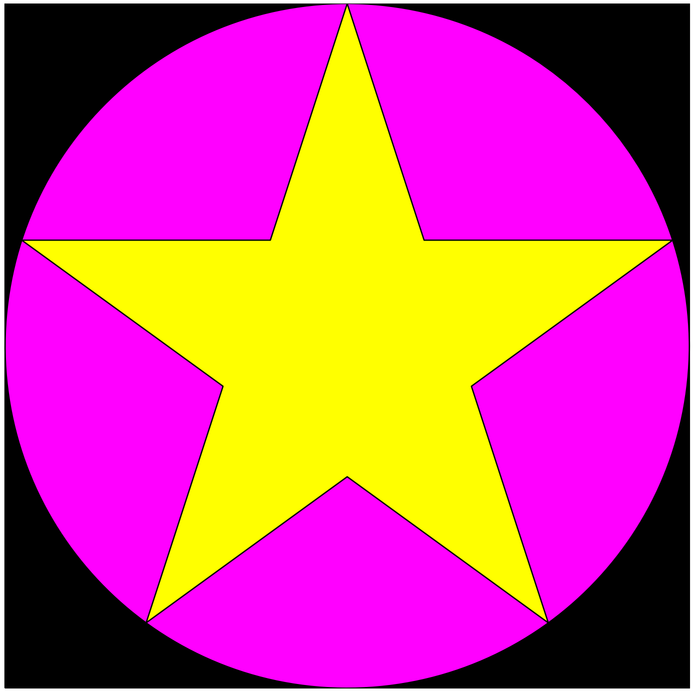


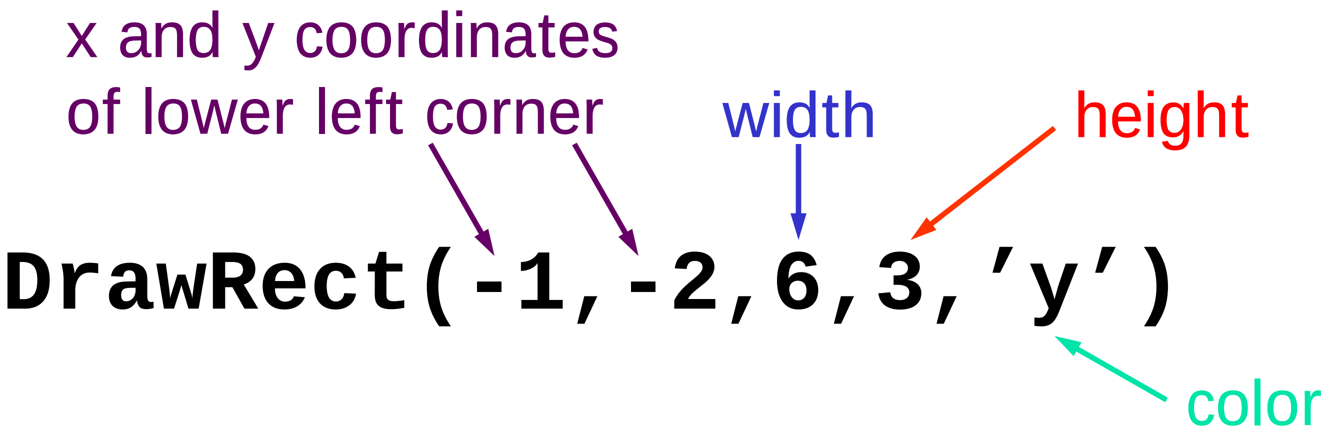
Review loops/conditionals using user-defined graphics function

Draw a black square;

then draw a magenta disk;

then draw a yellow star.





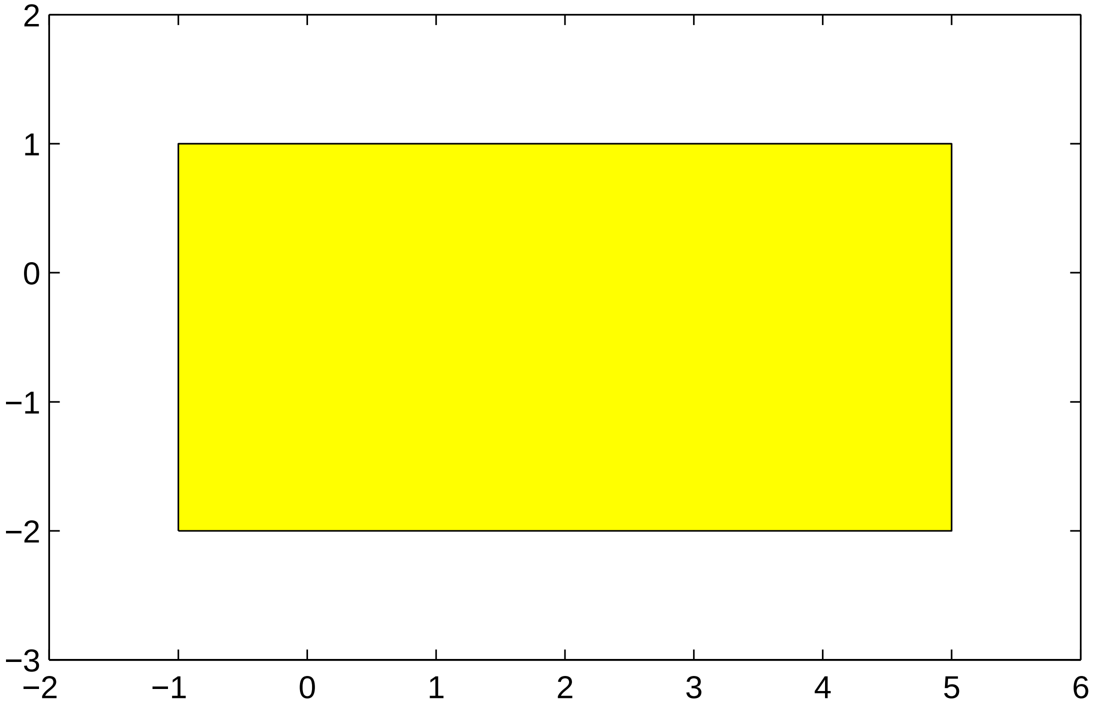
x and y coordinates
of lower left corner

width

height

```
DrawRect(-1,-2,6,3,’y’)
```

color

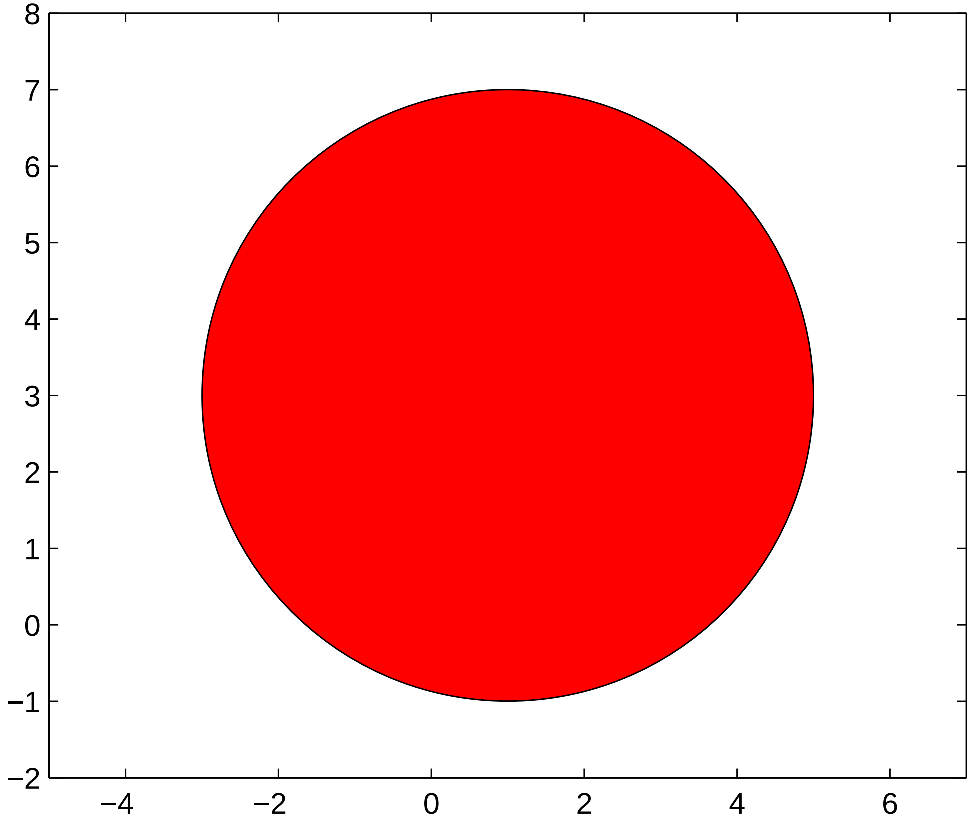


x and y coordinates of the center

radius

```
DrawDisk(1,3,4,'r')
```

color

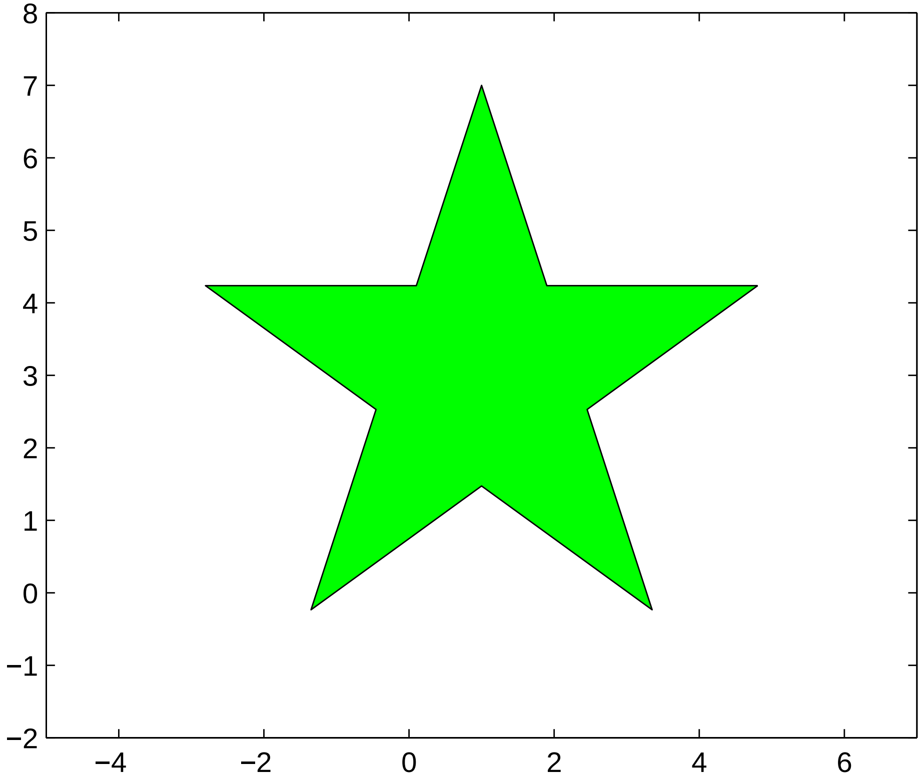


x and y coordinates of the center

“radius”

```
DrawStar(1,3,4,’g’)
```

color

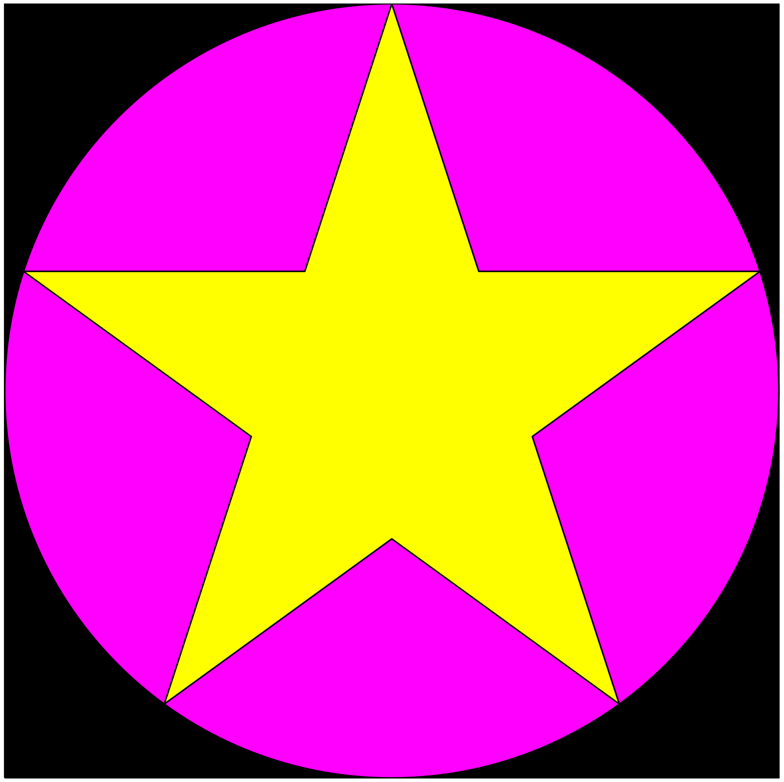


Color Options

White	`'w'`	⬜
Black	`'k'`	⬛
Red	`'r'`	🟥
Blue	`'b'`	🟦
Green	`'g'`	🟩
Yellow	`'y'`	🟨
Magenta	`'m'`	🟪
Cyan	`'c'`	🟦

A simple 3-line script

```
DrawRect(...)
DrawDisk(...)
DrawStar(...)
```



A general graphics framework

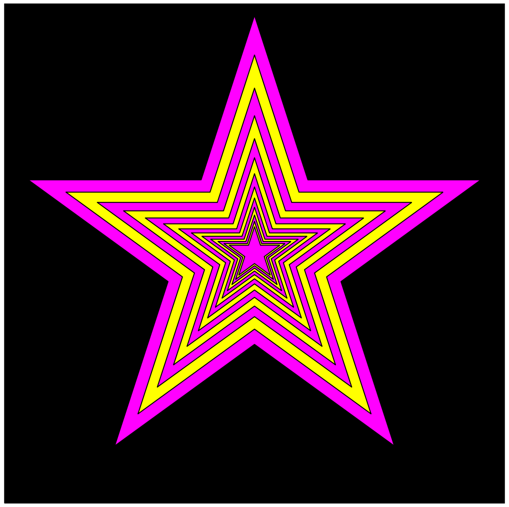
```
% drawDemo
close all
figure
axis equal off
hold on
```

Code fragment to draw the objects (rectangle, disk, star)

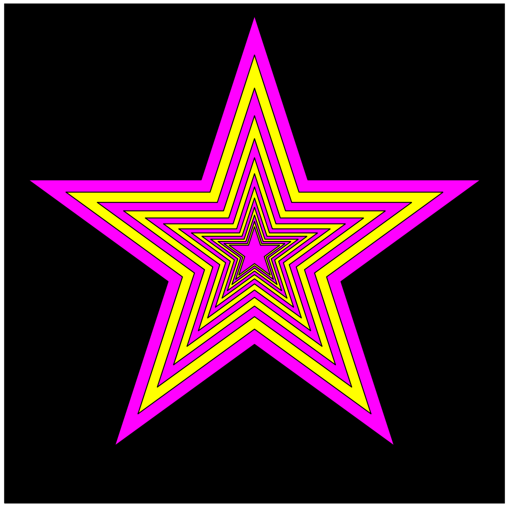
```
hold off
```

```
% drawDemo
close all
figure
axis equal off
hold on

DrawRect(0,0,2,2,'k')
DrawDisk(1,1,1,'m')
DrawStar(1,1,1,'y')

hold off
```

Example: Nested Stars



Example: Nested Stars



Draw a black square

-Bigger than the biggest star (at least 2 times radius of star)

- Center at (0,0)

Draw a sequence of stars

-Stars alternate in color

-Stars get smaller

  - radius r=1 to start

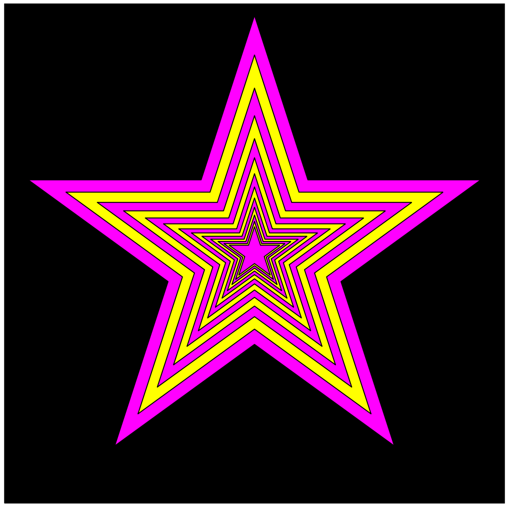
-1st star smaller than the sqr

-When to stop?

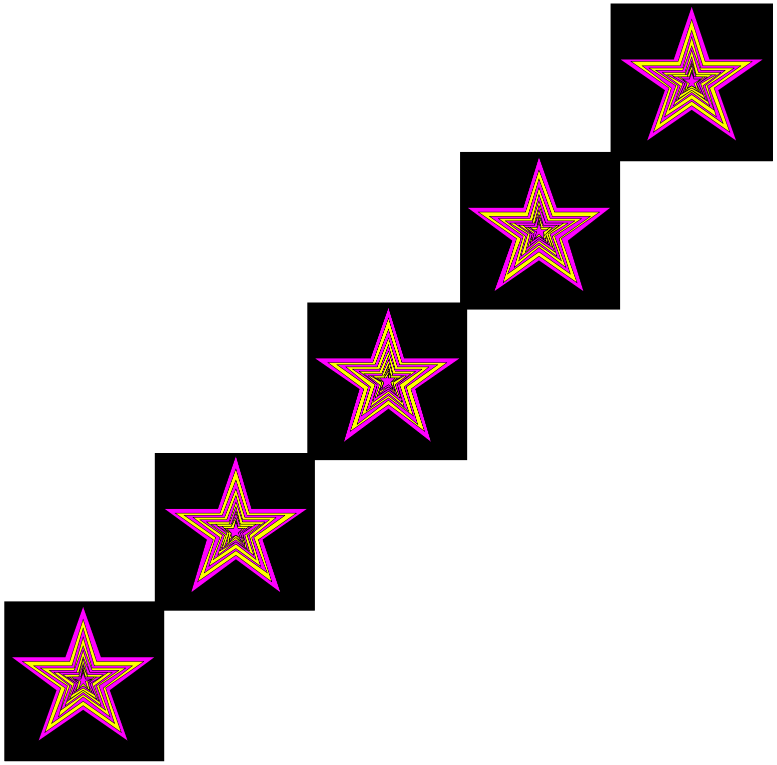
  - when r is small

nestedStars.m

Knowing how to draw



How difficult is it to draw



Pattern for doing something *n* times

```
n= ______
for k= 1:n

    % code to do
    % that something

end
```

```
x= 0; y= 0; % figure centered at (0,0)

s= 2.1;   % side length of square
DrawRect(x-s/2,y-s/2,s,s,’k’)

r= 1;  k= 1;
while  r > 0.1   %r still big
   % draw a star
   if rem(k,2)==1  %odd number
      DrawStar(x,y,r,’m’)  %magenta
   else
      DrawStar(x,y,r,’y’)  %yellow
   end
   % reduce r
   r= r/1.2;
   k= k + 1;
end
```

```
for c = 0:2:8
    x= c; y= c; % figure centered at (c,c)
    s= 2.1;   % side length of square
    DrawRect(x-s/2,y-s/2,s,s,’k’)

    r= 1;  k= 1;
    while  r > 0.1   %r still big
       % draw a star
       if rem(k,2)==1  %odd number
          DrawStar(x,y,r,’m’)  %magenta
       else
          DrawStar(x,y,r,’y’)  %yellow
       end
       % reduce r
       r= r/1.2;
       k= k + 1;
    end
end
```

Example: Are they prime?

- Given integers a and b, write a program that lists all the prime numbers in the range [a, b].
- Assume a>1, b>1 and a<b.

Example: Are they prime?
Subproblem: Is it prime?

- Given integers a and b, write a program that lists all the prime numbers in the range [a, b].
- Assume a>1, b>1 and a<b.
- Write a program fragment to determine whether a given integer n is prime, n>1.
- Reminder: rem(x,y) returns the remainder of x divided by y.

Example: Are they prime?

Subproblem: Is it prime?

```
for n = a:b
    % is n prime?
end
```

- Given integers a and b, write a program that lists all the prime numbers in the range [a, b].
- Assume a>1, b>1 and a<b.
- Write a program fragment to determine whether a given integer n is prime, n>1.
- Reminder: rem(x,y) returns the remainder of x divided by y.

```
Start :
    divisor = 2

Repeat :
    rem(n,divisor)
    divisor = divisor + 1

End:
    rem(n,divisor)==0
    divisor ≤ n ?
```

```
divisor = 2;

while (rem(n,divisor) ~= 0)
    divisor = divisor + 1;
end
if (divisor == n)
    disp('prime')
else
    disp('composite')
end
```

```
%Given n, display whether it is prime
divisor= 2;
while ( rem(n,divisor)~=0 )
     divisor= divisor + 1;
end
if (divisor==n)
     fprintf(‘%d is prime\n’, n)
else
     fprintf(‘%d is composite\n’, n)
end
```

```
for  n = a:b

    %Given n, display whether it is prime
    divisor= 2;
    while ( rem(n,divisor)~=0 )
         divisor= divisor + 1;
    end
    if (divisor==n)
         fprintf(‘%d is prime\n’, n)
    else
         fprintf(‘%d is composite\n’, n)
    end

end
```

Example: Times Table

Write a script to print a times table for a specified range.

Row headings

	3	4	5	6	7
3	9	12	15	18	21
4	12	16	20	24	28
5	15	20	25	30	35
6	18	24	30	36	42
7	21	28	35	42	49

Column headings

Developing the algorithm for the times table

	3	4	5	6	7
3	9	12	15	18	21
4	12	16	20	24	28
5	15	20	25	30	35
6	18	24	30	36	42
7	21	28	35	42	49

Developing the algorithm for the times table

	3	4	5	6	7
3	9	12	15	18	21
4	12	16	20	24	28
5	15	20	25	30	35
6	18	24	30	36	42
7	21	28	35	42	49

- Look for patterns
  - Each entry is row# × col#
  - Row#, col# increase regularly
- ⇒ Loop!!!
- What kind of loop?
  - for-loop—since the range of the headings will be specified and increment regularly
  - for each row#, get the products with all the col#s. Then go to next row# and get products with all col#s, …
  - ⇒ Nested loops!
- Details: what will be the print format? Don’t forget to start new lines. Also need initial input to specify the range.

```
disp('Show the times table for specified range')
lo= input('What is the lower bound? ');
hi= input('What is the upper bound? ');
```