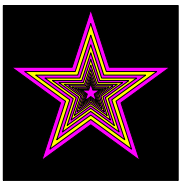
- Previous Lecture:
  - Iteration using **while**
- Today's Lecture:
  - Developing algorithms
  - Nested loops
- Announcements:
  - Section this week in the labs (Read FVL 3.2 before lab)
  - Project 2 due 2/14 (Thurs) at 6pm
  - Prelim 1 on 2/21 (Thurs) 7:30pm. If you have a conflict, tell us (email Kelly Patwell) now—no later than Wednesday evening.

Example: Nested Stars



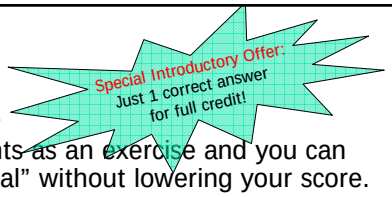
Draw a black square
- Bigger than the biggest star
- Center at (0,0)
- Make side length 2.1

Draw a sequence of stars
-Stars get smaller
-Stars alternate in color
-1st star smaller than the sqr
  -radius r=1 to start
-When to stop?
  - when r small

```
s= 2.1;  % side length of square
drawRect(-s/2,-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(0,0,r,'m')  %magenta
   else
      drawStar(0,0,r,'y')  %yellow
   end
   % reduce r
   r= r/1.2;
   k= k + 1;
end
```


Quiz 1



- 2 questions
- A quiz counts as an exercise and you can miss "several" without lowering your score. E.g., if there will be 15 exercises in total you can miss 3 (about 20%).
- Answer the quiz using your registered clicker.
- Honor system: Use only your clicker and don't consult your neighbors or notes in any way.

Example: Is it prime?

- Write a program fragment to determine whether a given integer n is prime. Assume n>1.
- Reminder: rem(x,y) returns the remainder of x divided by y.

Start:
  divisor = 2

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

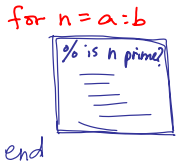
End:
  mod(n, divisor) == 0
  divisor ← n ?

```
divisor = 2;
foundyet = 0;
While ( ~foundyet )
  foundyet = (mod(n,d)==0);
  divisor = divisor + 1;
end
if (divisor-1 == n)
  disp('prime')
else
  disp('composite')
end
```

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


Example: Are they prime?

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



Example: Times Table

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

Row headings

Column 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

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 a specified range')
lo= input('What is the lower bound? ');
hi= input('What is the upper bound? ');

for r= lo:hi
   % at row r
   for c= lo:hi
      % at column c
      fprintf('%6d ', r*c)
   end
   fprintf('\n')
end
```


The savvy programmer...

- Learns useful programming patterns and use them where appropriate
- Seeks inspiration by working through test data "by hand"
  - Asks, "What am I doing?" at each step
  - Declares a variable for each piece of information maintained when working the problem by hand
- Decomposes the problem into manageable subtasks
  - Refines the solution iteratively, solving simpler subproblems first
- Remembers to check the problem's boundary conditions
- Validates the solution (program) by trying it on test data