- Previous Lecture:
  - Developing algorithms
  - Nested loops
- Today's Lecture:
  - Review nested loops
  - Finite vs. Infinite
  - Linear Interpolation
  - Introduction to vectors, vectorized code
- Announcements:
  - Project 2 due tonight 11pm
  - Prelim 1: 2/24(R) 7:30-9pm. Location Statler Aud.
  - Review sessions: next week, time and locations TBA. They're *optional*.



## Rational approximation of $\pi$

- $\pi$ = 3.141592653589793…
- Can be closely approximated by fractions, e.g., $\pi \approx 22/7$
- Rational number: a quotient of two integers
- Approximate $\pi$ as p/q where p and q are positive integers ≤M
- Start with a straight forward solution:
  - Get M from user
  - Calculate quotient p/q for all combinations of p and q
  - Pick best quotient → smallest error



```
% Rational approximation of pi

M = input('Enter M: ');


% Check all possible denominators
for q = 1:M

    For current q find best numerator p…
    Check all possible numerators



end
```



```
% Rational approximation of pi

M = input('Enter M: ');
% Best q, p, and error so far
qBest=1;  pBest=1;
err_pq = abs(pBest/qBest - pi);
% Check all possible denominators
for q = 1:M
  % Find best numerator for this q

  for p = 1:M  % Check all possible p


  end


end
myPi = pBest/qBest;
```



## Analyze the program for efficiency

- See Eg3_1 and FasterEg3_1 in the book

```
for a = 1:n
   disp('alpha')
   for b = 1:m
      disp('beta')
   end
end
```

How many times are "alpha" and "beta" displayed?

A: n, m

B: m, n

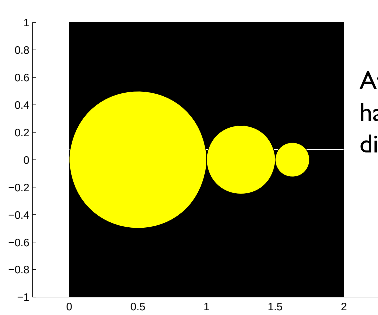
C: n, n+m

D: n, n*m

E: m*n, m



## Screen Granularity




After how many halvings will the disks disappear?

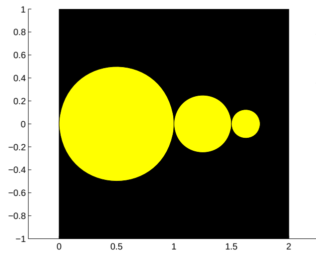
## Xeno's Paradox

- A wall is two feet away
- Take steps that repeatedly halve the remaining distance
- You never reach the wall because the distance traveled after n steps =

$$1 + \frac{1}{2} + \frac{1}{4} + \ldots + 1/2^n = 2 - 1/2^n$$



## Example: "Xeno" disks



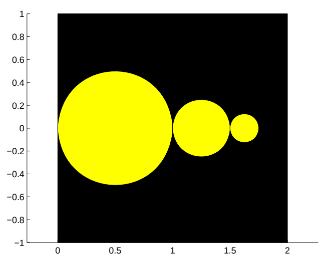

Draw a sequence of 20 disks where the (k+1)th disk has a diameter that is half that of the kth disk.

The disks are tangent to each other and have centers on the x-axis.

First disk has diameter 1 and center (1/2, 0).



## Example: "Xeno" disks




What do you need to keep track of?

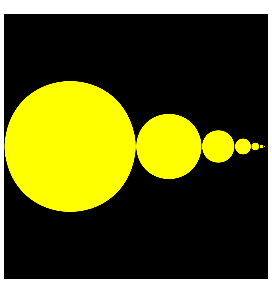
- Diameter (d)
- Position
  Left tangent point (x)

| Disk | x | d |
|---|---|---|
| 1 | 0 | 1 |
| 2 | 0+1 | 1/2 |
| 3 | 0+1+1/2 | 1/4 |



```
% Xeno Disks

DrawRect(0,-1,2,2,'k')
% Draw 20 Xeno disks


for k= 1:20






end
```

## Here's the output… Shouldn't there be 20 disks?



The "screen" is an array of dots called pixels.

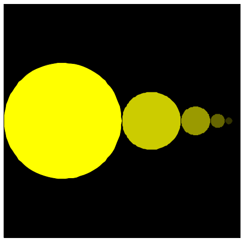
Disks smaller than the dots don't show up.

The 20th disk has radius<.000001



*End of Material for Prelim 1*

## Fading Xeno disks



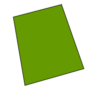
- First disk is yellow
- Last disk is black (invisible)
- *Interpolate* the color in between



## Color is a 3-vector, sometimes called the RGB values

- Any color is a mix of red, green, and blue
- Example:

```
colr= [0.4  0.6  0]
```



- Each component is a real value in [0,1]
- [0 0 0] is black
- [1 1 1] is white



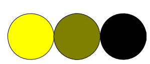
## Example: 3 disks fading from yellow to black

```
r= 1;  % radius of  disk
yellow= [1 1 0];
black = [0 0 0];

% Left disk yellow, at x=1
DrawDisk(1,0,r,yellow)
% Right disk black, at x=5
DrawDisk(5,0,r,black)

% Middle disk with average color, at x=3
colr= 0.5*yellow + 0.5*black;
DrawDisk(3,0,r,colr)
```

.5 * [1 1 0] → [.5 .5 0]

.5 * [0 0 0] → [0 0 0]

Vectorized multiplication





## Linear interpolation

| x | g(x) |
|---|---|
| : | : |
| 9 | 110 |
| 10 | 118 |
| 10.25 | ? |
| 10.50 | ? |
| 10.75 | ? |
| 11 | 126 |
| 12 | 134 |
| : | : |

$g(10.5) = \frac{1}{2}\,g(11) + \frac{1}{2}\,g(10)$

$g(10.25) = 1/4 \cdot g(11) + 3/4 \cdot g(10)$
$g(10.50) = 2/4 \cdot g(11) + 2/4 \cdot g(10)$
$g(10.75) = 3/4\ g(11) + 1/4\ g(10)$

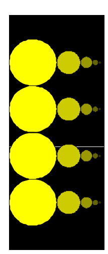


```
% Draw n fading Xeno disks
d= 1;
x= 0;  % Left tangent point
yellow= [1 1 0];
black=  [0 0 0];
for k= 1:n
   % Compute color of kth disk
   f= ???
   colr= f*black + (1-f)*yellow;
   % Draw kth disk
   DrawDisk(x+d/2, 0, d/2, colr)
   x= x+d;
   d= d/2;
end
```

A k/n
B k/(n-1)
C (k-1)/n
D (k-1)/(n-1)
E (k-1)/(n+1)



## Rows of Xeno disks



```
for y = __ : __ : __

     Code to draw one
     row of Xeno disks
     at some y-coordinate

end
```

Be careful with "initializations"

## Vectorized subtraction

| | | | | | |
|---|---|---|---|---|---|
| | x | 2 | 1 | .5 | 8 |
| - | y | 1 | 2 | 0 | 1 |
| = | z | 1 | -1 | .5 | 7 |

Matlab code: `z= x - y`

Lecture 8 54

## Vectorized code
—a Matlab-specific feature

See Sec 4.1 for list of vectorized arithmetic operations

- Code that performs element-by-element arithmetic/relational/logical operations on array operands in one step
- Scalar operation: x + y
  where x, y are scalar variables
- Vectorized code: x + y
  where x and/or y are vectors. If x and y are both vectors, they must be of the same shape and length

Lecture 8 55

## Vectorized multiplication

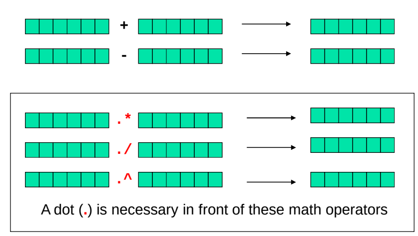
| | | | | | |
|---|---|---|---|---|---|
| | a | 2 | 1 | .5 | 8 |
| × | b | 1 | 2 | 0 | 1 |
| = | c | 2 | 2 | 0 | 8 |

Matlab code: `c= a .* b`

Lecture 8 56

## Vectorized element-by-element arithmetic operations on arrays

See full list of ops in §4.1




Lecture 8 57

## Reciprocate

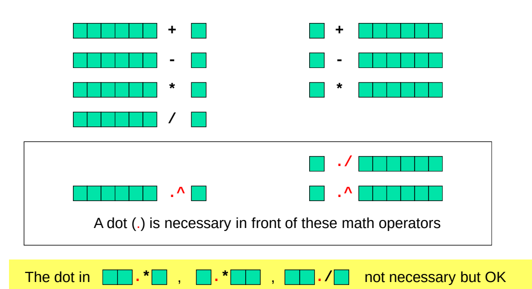
| | | | | | |
|---|---|---|---|---|---|
| | x | 1 | | | |
| / | y | 2 | 1 | .5 | 8 |
| = | z | .5 | 1 | 2 | .125 |

Matlab code: `z= x ./ y`

Lecture 8 59

## Vectorized element-by-element arithmetic operations between an array and a scalar

See full list of ops in §4.1




Lecture 8 60