

- Previous Lecture:
 - Examples on vectors and simulation
- Today's Lecture:
 - Finite vs. Infinite; Discrete vs. Continuous
 - Vectors and vectorized code
 - Color computation with linear interpolation
 - `plot` and `fill`
- Announcements:
 - `Project 3` due Friday 10/3 at 11pm
 - `Prelim I` on Oct 16th at 7:30pm. Email Randy Hess (rbh27) now if you have an exam conflict (specify conflicting course and instructor contact info)

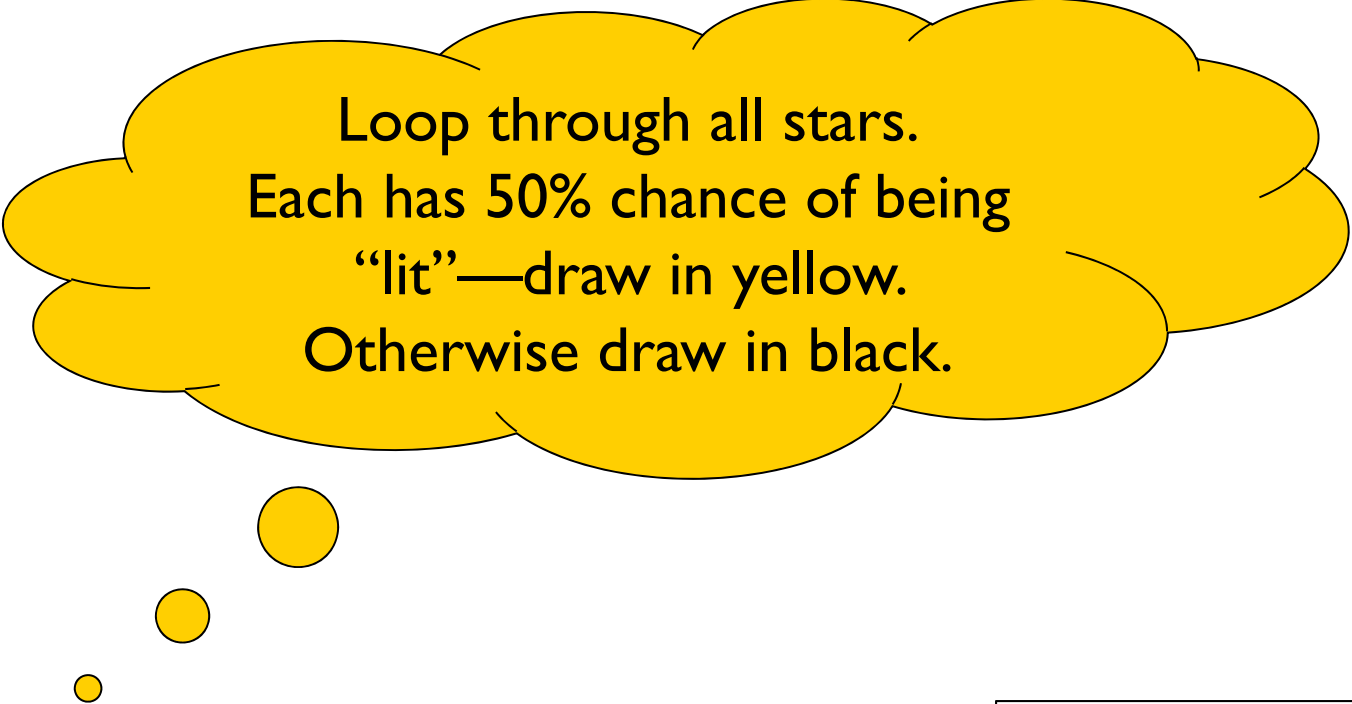
Simulate twinkling stars

- Get 10 user mouse clicks as locations of 10 stars—our constellation
- Simulate twinkling
 - Loop through all the stars; each has equal likelihood of being bright or dark
 - Repeat many times
- Can use DrawStar, DrawRect

```
% No. of stars and star radius
N=10;  r=.5;
% Get mouse clicks, store coords in vectors x,y
[x,y] = ginput(N);
% Twinkle!
for k= 1:20  % 20 rounds of twinkling

end
```

```
% No. of stars and star radius
N=10;  r=.5;
% Get mouse clicks, store coords In vectors x,y
[x,y] = ginput(N);
% Twinkle!
for k= 1:20  % 20 rounds of twinkling
```

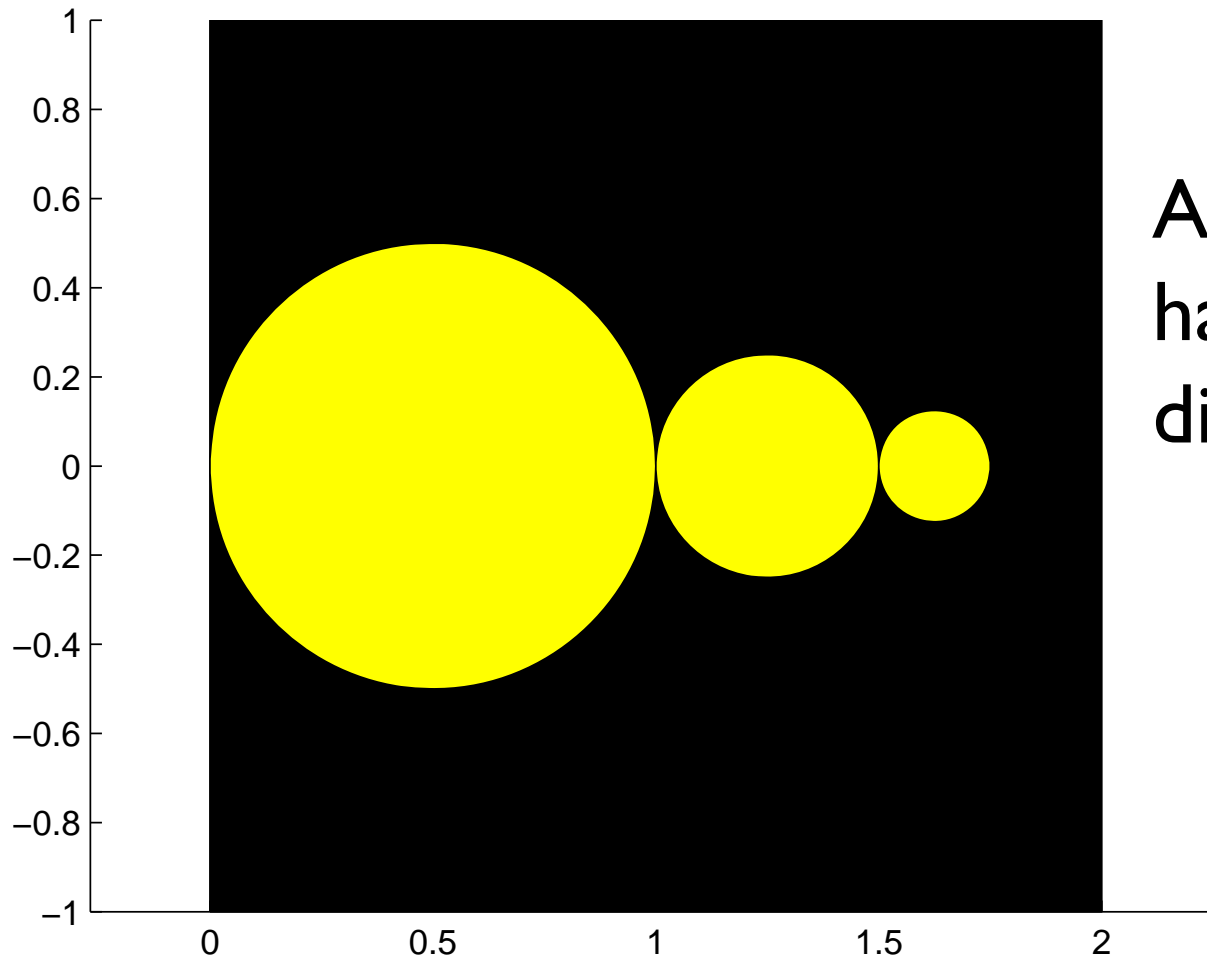


Loop through all stars.
Each has 50% chance of being
“lit”—draw in yellow.
Otherwise draw in black.

```
end
```

[See Twinkle.m](#)

Screen Granularity



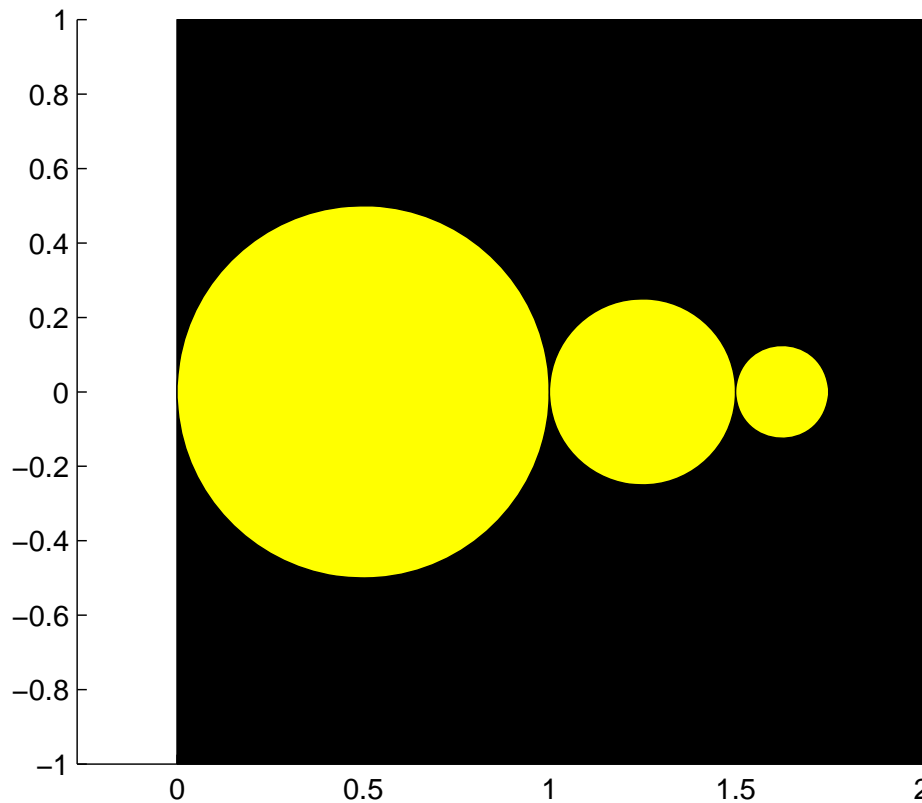
After how many halvings will the disks disappear?

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 =

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

Example: “Xeno” disks

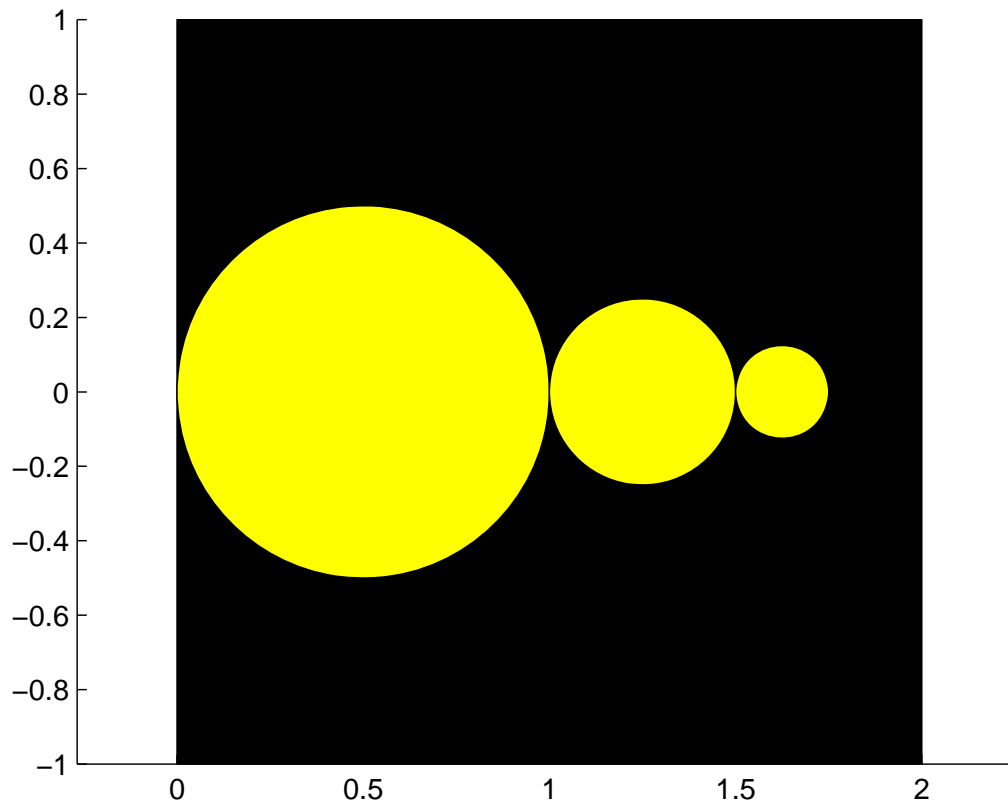


Draw a sequence of 20 disks where the $(k+1)$ th disk has a diameter that is half that of the k th disk.

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
```

```
% Xeno Disks
```

```
DrawRect(0,-1,2,2,'k')
```

```
% Draw 20 Xeno disks
```

```
for k= 1:20
```

```
end
```

```
% Xeno Disks
```

```
DrawRect(0,-1,2,2,'k')
```

```
% Draw 20 Xeno disks
```

```
d= 1;
```

```
x= 0; % Left tangent point
```

```
for k= 1:20
```

```
end
```

```
% Xeno Disks
```

```
DrawRect(0,-1,2,2,'k')
```

```
% Draw 20 Xeno disks
```

```
d= 1;
```

```
x= 0; % Left tangent point
```

```
for k= 1:20
```

```
    % Draw kth disk
```

```
    % Update x, d for next disk
```

```
end
```

```
% Xeno Disks
```

```
DrawRect(0,-1,2,2,'k')
```

```
% Draw 20 Xeno disks
```

```
d= 1;
```

```
x= 0; % Left tangent point
```

```
for k= 1:20
```

```
    % Draw kth disk
```

```
        DrawDisk(x+d/2, 0, d/2, 'y')
```

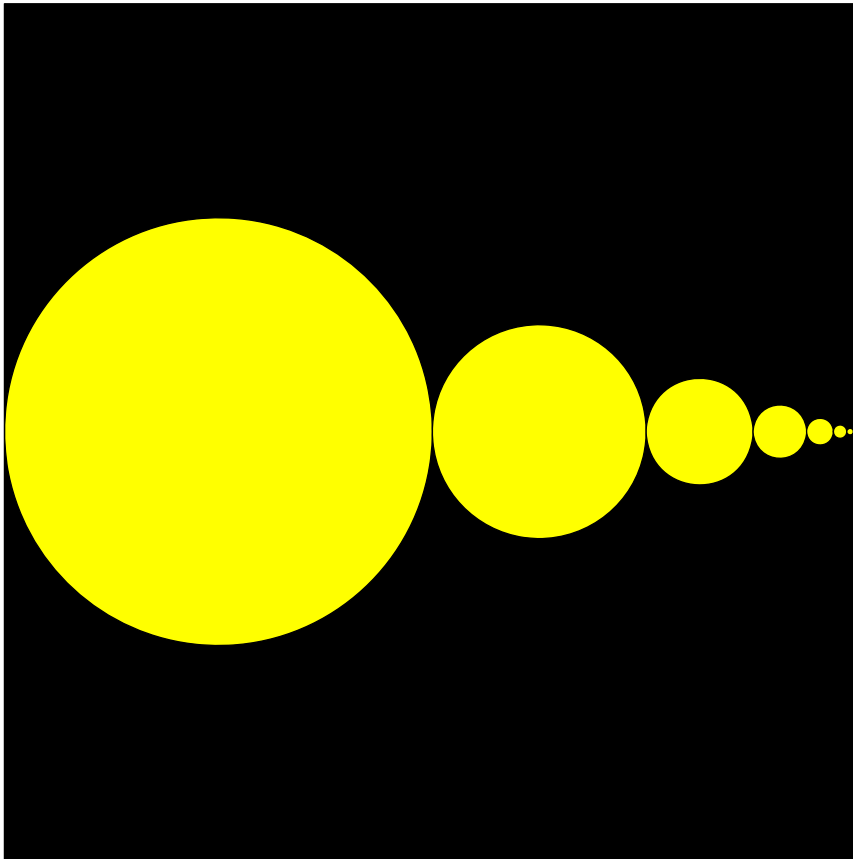
```
    % Update x, d for next disk
```

```
        x= x+d;
```

```
        d= d/2;
```

```
end
```

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

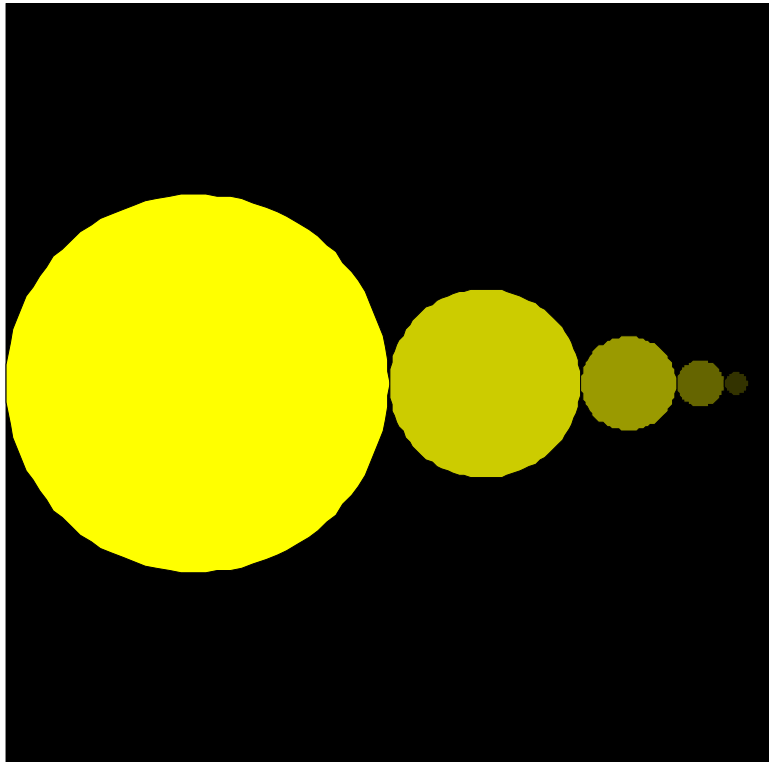


The “screen” is an array of dots called pixels.

Disks smaller than the dots don't show up.

The 20th disk has
radius $< .000001$

Fading Xeno disks



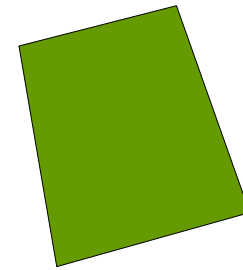
- First disk is **yellow**
- Last disk is black (invisible)
- Interpolate the color in between

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

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

- Example:

`color = [0.4 0.6 0]`



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


```
% Draw n Xeno disks
d= 1;
x= 0; % Left tangent point

for k= 1:n

    % Draw kth disk
    DrawDisk(x+d/2, 0, d/2, 'y')
    x= x+d;
    d= d/2;
end
```

```
% Draw n Xeno disks
d= 1;
x= 0; % Left tangent point
```

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

```
    % Draw kth disk
```

```
    DrawDisk(x+d/2, 0, d/2, [1 1 0])
```

```
    x= x+d;
```

```
    d= d/2;
```

```
end
```

A vector of length 3



```

% 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

    % Draw kth disk
    DrawDisk(x+d/2, 0, d/2, _____)
    x= x+d;
    d= d/2;
end

```

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)
```

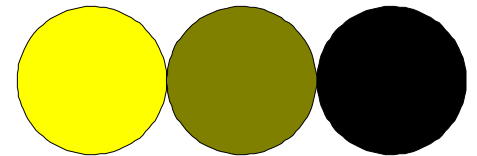
```
% Right disk black, at x=5
```

```
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)
```



Example: 3 disks fading from yellow to black

```
r= 1; % radius of disk
```

```
yellow= [1 1 0];
```

```
black = [0 0 0];
```

$$\begin{array}{|c|} \hline .5 \\ \hline \end{array} * \begin{array}{|c|c|c|} \hline 1 & 1 & 0 \\ \hline \end{array} \longrightarrow \begin{array}{|c|c|c|} \hline .5 & .5 & 0 \\ \hline \end{array}$$

$$\begin{array}{|c|} \hline .5 \\ \hline \end{array} * \begin{array}{|c|c|c|} \hline 0 & 0 & 0 \\ \hline \end{array} \longrightarrow \begin{array}{|c|c|c|} \hline 0 & 0 & 0 \\ \hline \end{array}$$

```
% Left disk yellow, at x=1
```

```
DrawDisk(1,0,r,yellow)
```

```
% Right disk black, at x=5
```

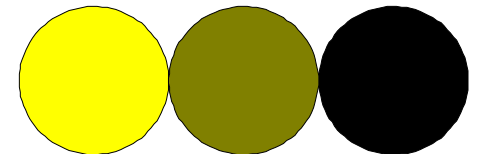
```
DrawDisk(5,0,r,black)
```

```
% Middle disk with average color, at x=3
```

```
colr= 0.5*yellow + 0.5*black;
```

```
DrawDisk(3,0,r,colr)
```

Vectorized
multiplication



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)
```

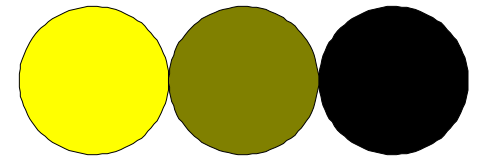
```
% Middle disk with average color, at x=3
```

```
colr= 0.5*yellow + 0.5*black;
```

```
DrawDisk(3,0,r,colr)
```

Vectorized
addition

$$\begin{array}{r} \begin{array}{|c|c|c|} \hline .5 & .5 & 0 \\ \hline \end{array} \\ + \\ \begin{array}{|c|c|c|} \hline 0 & 0 & 0 \\ \hline \end{array} \\ \hline = \\ \begin{array}{|c|c|c|} \hline .5 & .5 & 0 \\ \hline \end{array} \end{array}$$



Vectorized code allows an operation on multiple values at the same time

```
yellow= [1 1 0];  
black = [0 0 0];
```

Vectorized
addition

$$\begin{array}{|c|c|c|} \hline .5 & .5 & 0 \\ \hline \end{array} + \begin{array}{|c|c|c|} \hline 0 & 0 & 0 \\ \hline \end{array} \\ \hline = \begin{array}{|c|c|c|} \hline .5 & .5 & 0 \\ \hline \end{array}$$

% Average color via vectorized op

```
colr= 0.5*yellow + 0.5*black;
```

Operation performed on vectors

% Average color via scalar op

```
for k = 1:length(black)
```

```
    colr(k)= 0.5*yellow(k) + 0.5*black(k);
```

```
end
```

Operation performed on scalars

```

% 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

    % Draw kth disk
    DrawDisk(x+d/2, 0, d/2, _____)
    x= x+d;
    d= d/2;
end

```



```

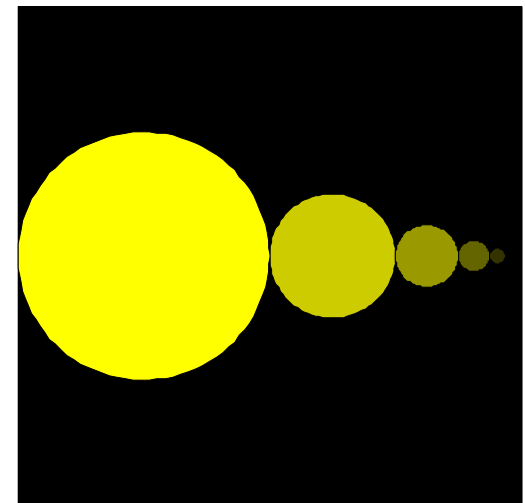
% Draw n fading Xeno disks
d= 1;
x= 0; % Left tangent point
yellow= [1 1 0];
black= [0 0 0];
for k= 1:n
    % Compute color of kth disk
    f= ???
    colr= f*black + (1-f)*yellow;
    % Draw kth disk
    DrawDisk(x+d/2, 0, d/2, colr)
    x= x+d;
    d= d/2;
end

```

Use linear interpolation to obtain the colors. Each disk has a color that is a linear combination of yellow and black. Let f be a fraction in $(0,1)$...

```
f= ???
```

```
color= f*black + (1-f)*yellow;
```



Linear interpolation

x	$g(x)$
:	:
9	110
10	118
11	126
12	134
:	:

Linear interpolation

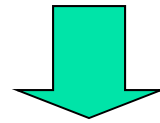
x	g(x)
:	:
9	110
10	118
10.25	?
10.50	?
10.75	?
11	126
12	134
:	:

$$g(10.5) = [g(11) + g(10)] / 2$$

Linear interpolation

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

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



$$g(10.25) = \frac{1}{4} \cdot g(11) + \frac{3}{4} \cdot g(10)$$

$$g(10.50) = \frac{2}{4} \cdot g(11) + \frac{2}{4} \cdot g(10)$$

$$g(10.75) = \frac{3}{4} \cdot g(11) + \frac{1}{4} \cdot g(10)$$

Linear interpolation

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

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

$$g(10) = \frac{0}{4} \cdot g(11) + \frac{4}{4} \cdot g(10)$$

$$g(10.25) = \frac{1}{4} \cdot g(11) + \frac{3}{4} \cdot g(10)$$

$$g(10.50) = \frac{2}{4} \cdot g(11) + \frac{2}{4} \cdot g(10)$$

$$g(10.75) = \frac{3}{4} \cdot g(11) + \frac{1}{4} \cdot g(10)$$

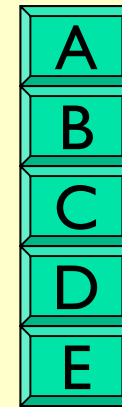
$$g(11) = \frac{4}{4} \cdot g(11) + \frac{0}{4} \cdot g(10)$$

$$f \cdot g(11) + (1-f) \cdot g(10)$$

```

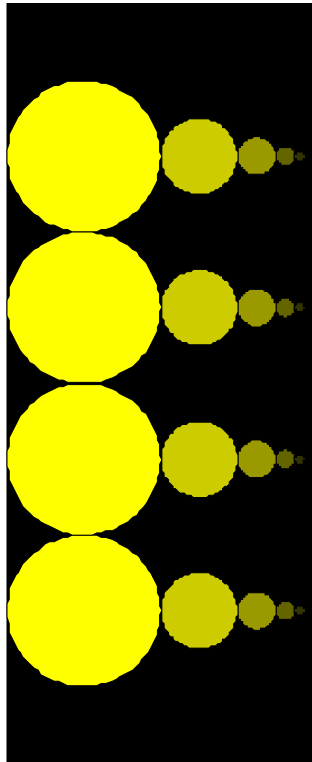
% Draw n fading Xeno disks
d= 1;
x= 0; % Left tangent point
yellow= [1 1 0];
black= [0 0 0];
for k= 1:n
    % Compute color of kth disk
    f= ???
    colr= f*black + (1-f)*yellow;
    % Draw kth disk
    DrawDisk(x+d/2, 0, d/2, colr)
    x= x+d;
    d= d/2;
end

```



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"


```
yellow=[1 1 0];  black=[0 0 0];

d= 1;

x= 0;

for k= 1:n
    % Compute color of kth disk
    f= (k-1)/(n-1);
    colr= f*black + (1-f)*yellow;
    % Draw kth disk
    DrawDisk(x+d/2, 0, d/2, colr)
    x=x+d;  d=d/2;
end
```

Where to put the loop header for `y=__:__:__`

A →

```
yellow=[1 1 0]; black=[0 0 0];
```

B →

```
d= 1;
```

C →

```
x= 0;
```

D →

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

```
    % Compute color of kth disk
```

```
    f= (k-1)/(n-1);
```

```
    colr= f*black + (1-f)*yellow;
```

```
    % Draw kth disk
```

```
    DrawDisk(x+d/2, 0, d/2, colr)
```

```
    x=x+d; d=d/2;
```

```
end
```

```
end
```

y

```

    yellow=[1 1 0];  black=[0 0 0];
for y= ___:___:___
    d= 1;
    x= 0;
    for k= 1:n
        % Compute color of kth disk
        f= (k-1)/(n-1);
        colr= f*black + (1-f)*yellow;
        % Draw kth disk
        DrawDisk(x+d/2, 0, d/2, colr)
        x=x+d;  d=d/2;
    end
end

```

initializations necessary for each row

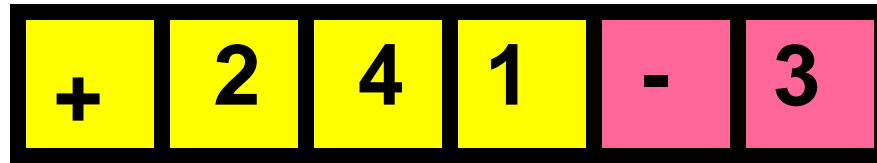
y

Does this script print anything?

```
k = 0;  
while 1 + 1/2^k > 1  
    k = k+1;  
end  
disp(k)
```

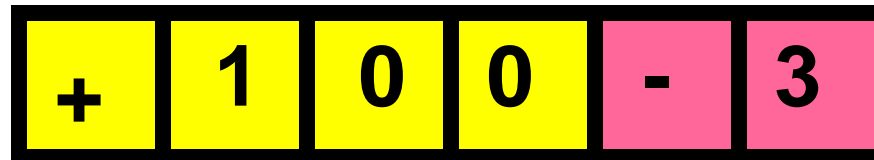
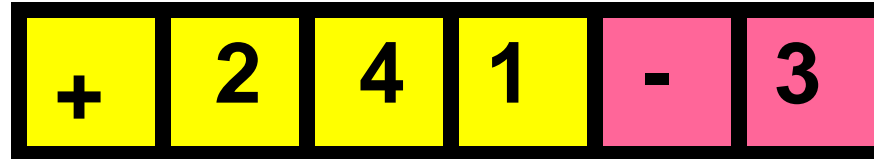
Computer Arithmetic—floating point arithmetic

Suppose you have a calculator with a window like this:

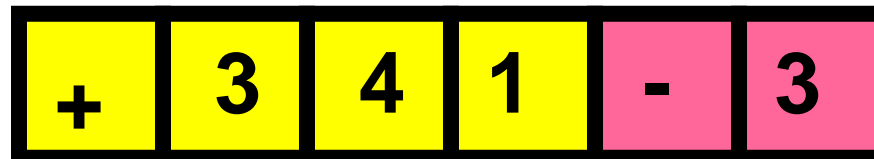


representing 2.41×10^{-3}

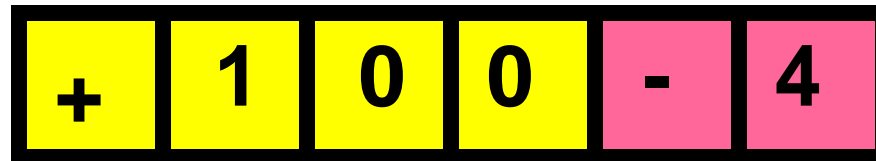
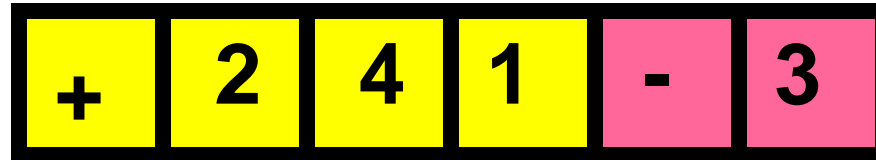
Floating point addition



Result:



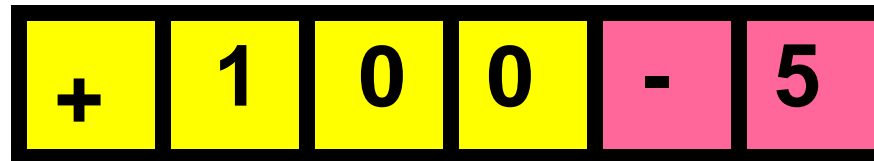
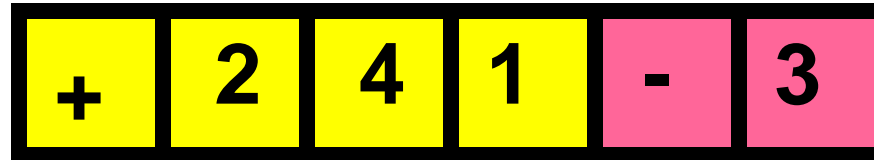
Floating point addition



Result:



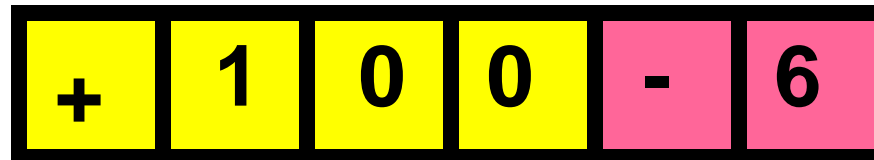
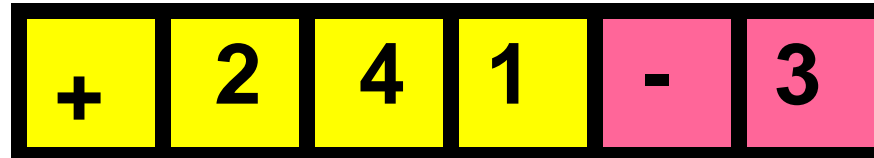
Floating point addition



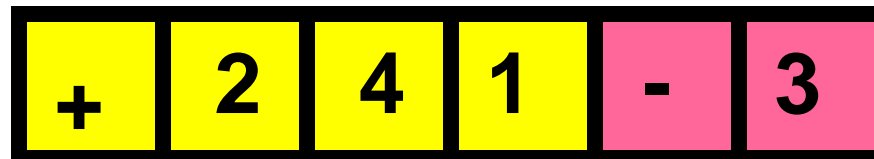
Result:



Floating point addition



Result:



Floating point addition

+	2	4	1	-	3
---	---	---	---	---	---

+	1	0	0	-	6
---	---	---	---	---	---

Result:

+	2	4	1	-	3
---	---	---	---	---	---

Not enough room to represent .002411

The loop DOES terminate given the limitations of floating point arithmetic!

```
k = 0;  
while 1 + 1/2^k > 1  
    k = k+1;  
end  
disp(k)
```

$1 + 1/2^{53}$ is calculated to be just 1,
so "53" is printed.

Patriot missile failure



www.namsa.nato.int/gallery/systems

In 1991, a Patriot Missile failed, resulting in 28 deaths and about 100 injured. The cause?

0.1

Inexact representation of time/number

- System clock represented time in tenths of a second: a clock tick every 1/10 of a second

- Time = number of clock ticks $\times 0.1$

"exact" value

.000110011001100110011001100110011...

.00011001100110011001100110011 value in Patriot system

Error of .000000095 every clock tick

Resulting error

... after 100 hours

$$.000000095 \times (100 \times 60 \times 60)$$

0.34 second

At a velocity of 1700 m/s, missed target by more than 500 meters!

