- 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 π

- π = 3.141592653589793…
- Can be closely approximated by fractions,
  e.g., $\pi \approx 22/7$
- Rational number: a quotient of two integers
- Approximate π 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
```

```
% Rational approximation of pi

M = input('Enter M: ');



% Check all possible denominators
for q = 1:M









end
```

```
% 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: ');



% Check all possible denominators
for q = 1:M
  % Find best numerator for this q

  for p = 1:M  % Check all possible p



  end




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;
```

```
% 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;
```

```
% 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
  p0=1;  e0=abs(p0/q - pi);  % best p & error so far
  for p = 1:M  % Check all possible p




  end




end
myPi = pBest/qBest;
```

```
% 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
  p0=1;  e0=abs(p0/q - pi);  % best p & error so far
  for p = 1:M  % Check all possible p
    if abs(p/q - pi) < e0  % new best numerator found
      p0=p;  e0 = abs(p0/q - pi);
    end
  end
```

Now we have the best p for this q.
Is the quotient at this q best among all previous q's?

```
end
myPi = pBest/qBest;
```

```
% 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
  p0=1;  e0=abs(p0/q - pi);  % best p & error so far
  for p = 1:M  % Check all possible p
    if abs(p/q - pi) < e0  % new best numerator found
      p0=p;  e0 = abs(p0/q - pi);
    end
  end
```

Now we have the best p for this q.
Is the quotient at this q best among all previous q's?

```
end
myPi = pBest/qBest;
```

```
% 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
  p0=1;  e0=abs(p0/q - pi);  % best p & error so far
  for p = 1:M  % Check all possible p
    if abs(p/q - pi) < e0  % new best numerator found
      p0=p;  e0 = abs(p0/q - pi);
    end
  end
  % Is best quotient for this q is best over all?
  if e0 < err_pq
    pBest=p0;  qBest=q; err_pq=e0;
  end
end
myPi = pBest/qBest;
```

Analyze the program for efficiency

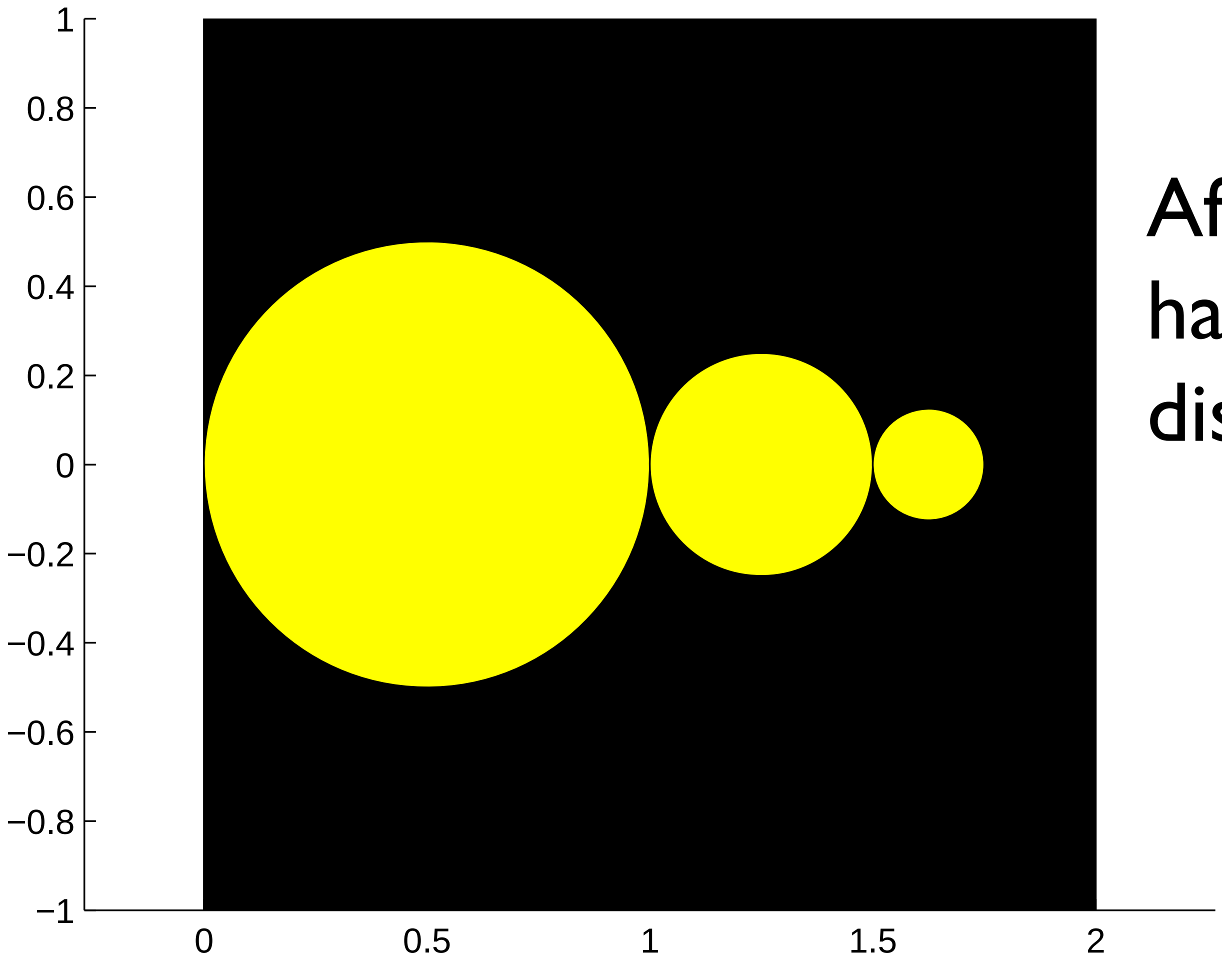
- See Eg3_1 and FasterEg3_1 in the book

```
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



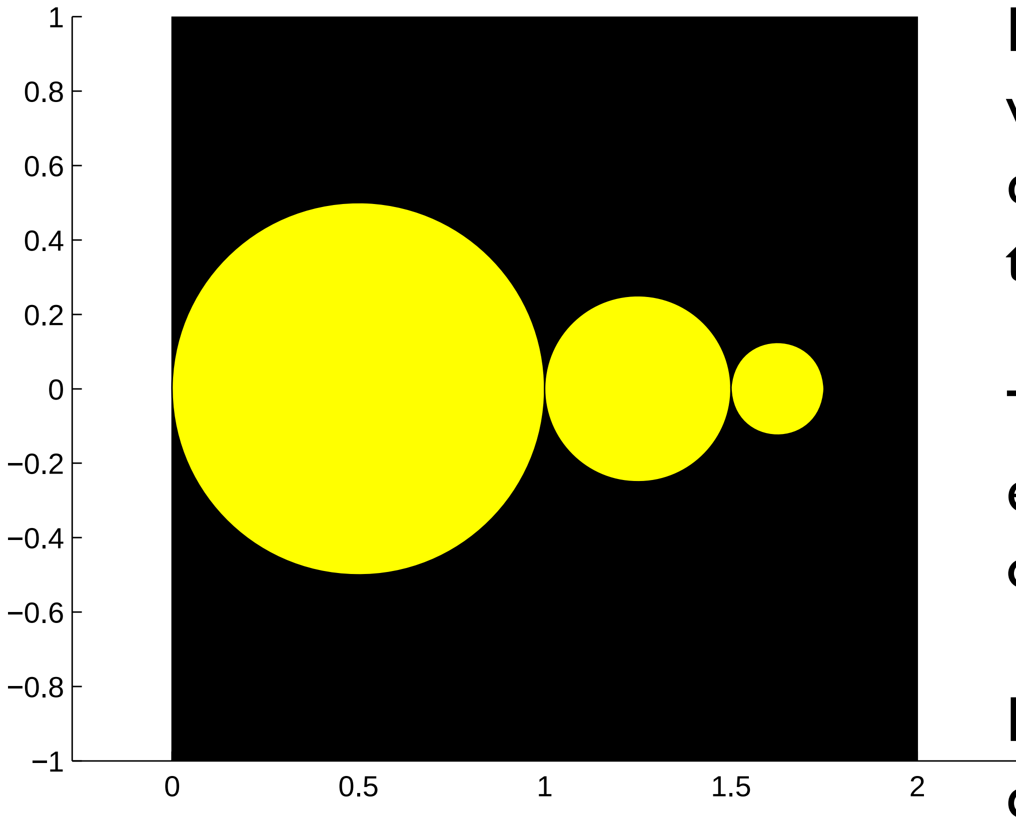
After how many halvings will the disks disappear?

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 =

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

Example: “Xeno” disks

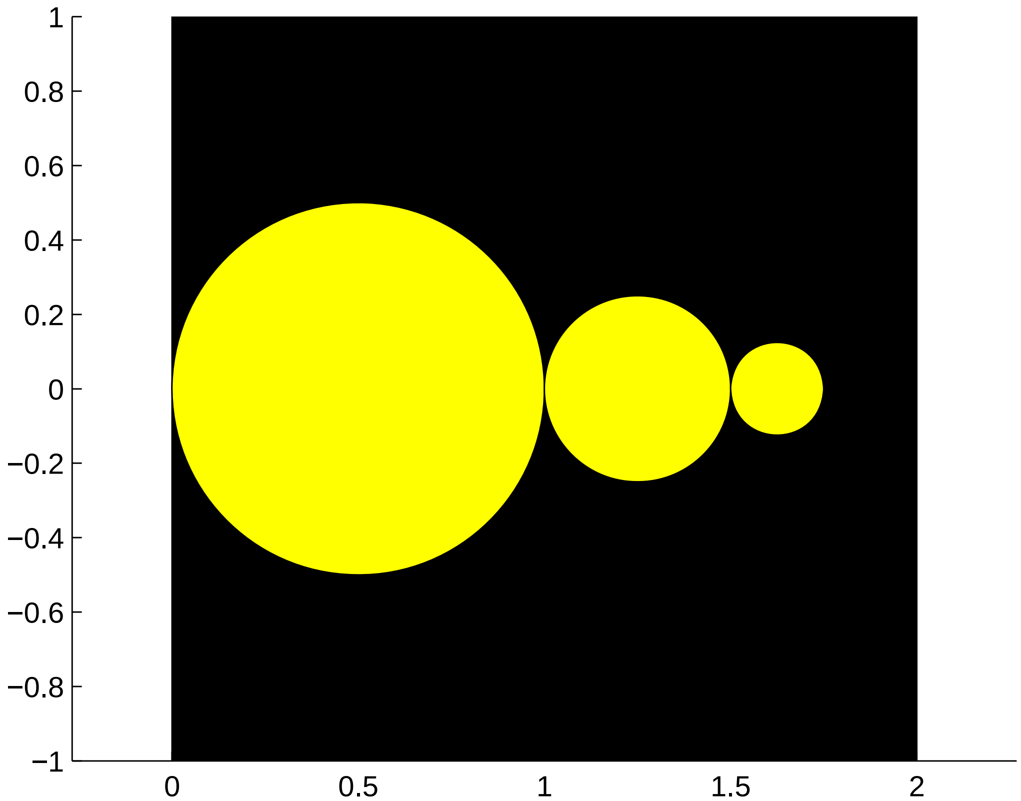


Draw a sequence of 20 disks where the (k+1)th disk has a diameter that is half that of the kth disk.

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
```

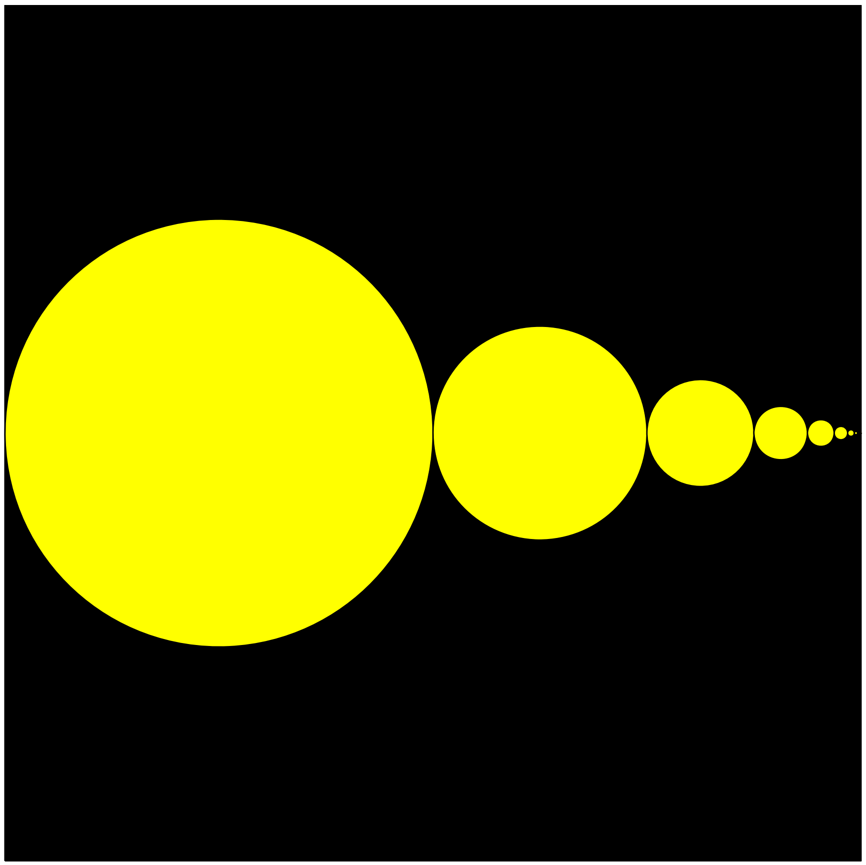
```
% Xeno Disks

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?



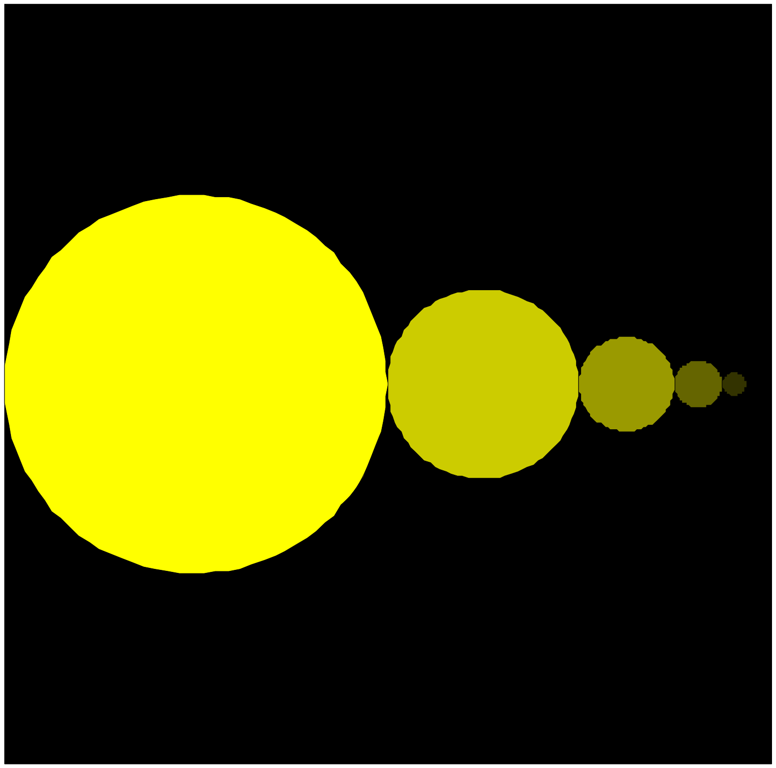
The "screen" is an array of dots called pixels.

Disks smaller than the dots don't show up.

The 20th disk has radius<.000001

End of Material for Prelim 1

Fading Xeno disks

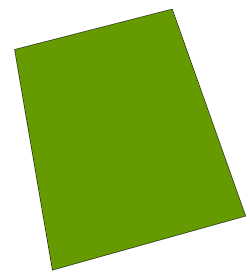


- First disk is yellow
- Last disk is black (invisible)
- *Interpolate* the color in between

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

```
% 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
```

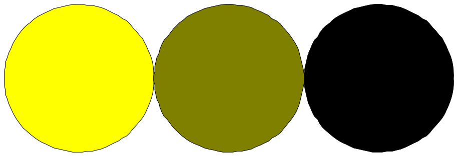
```
% Draw n Xeno disks
d= 1;
x= 0;  % Left tangent point


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

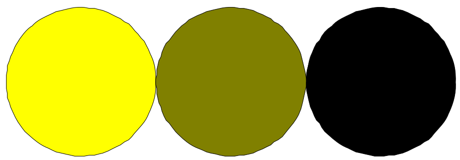
```
r= 1;  % radius of  disk
yellow= [1 1 0];
black = [0 0 0];
```

$$.5 * \begin{bmatrix} 1 & 1 & 0 \end{bmatrix} \rightarrow \begin{bmatrix} .5 & .5 & 0 \end{bmatrix}$$

$$.5 * \begin{bmatrix} 0 & 0 & 0 \end{bmatrix} \rightarrow \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}$$

Vectorized multiplication

```
% 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)
```

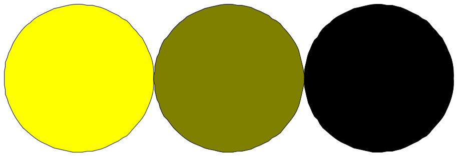


Example: 3 disks fading from yellow to black

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

% 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

$$
\begin{array}{cc}
 & \boxed{.5}\boxed{.5}\boxed{0} \\
+ & \boxed{0}\boxed{0}\boxed{0} \\
\hline
= & \boxed{.5}\boxed{.5}\boxed{0}
\end{array}
$$



```
% 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
```

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
:	:

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

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

Linear interpolation

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

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

$$\begin{aligned}
g(10) &= 0/4 \cdot g(11) + 4/4 \cdot g(10) \\
g(10.25) &= 1/4 \cdot g(11) + 3/4 \cdot g(10) \\
g(10.50) &= 2/4 \cdot g(11) + 2/4 \cdot g(10) \\
g(10.75) &= 3/4 \cdot g(11) + 1/4 \cdot g(10) \\
g(11) &= 4/4 \cdot g(11) + 0/4 \cdot g(10)
\end{aligned}$$

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

```
% 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



   % 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
```

A k/n

B k/(n-1)

C (k-1)/n

D (k-1)/(n-1)

E (k-1)/(n+1)

Linear interpolation

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

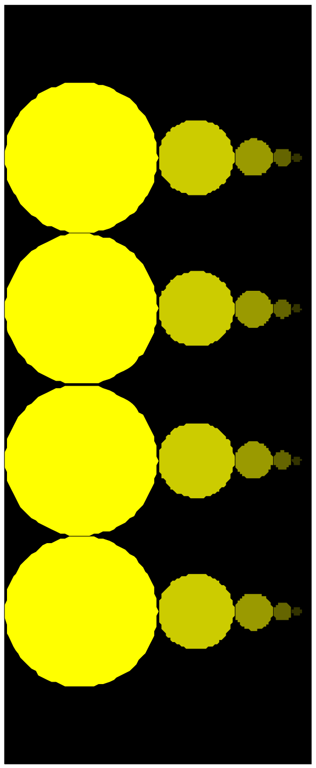
$$g(10) = 0/4 \cdot g(11) + 4/4 \cdot g(10)$$
$$g(10.25) = 1/4 \cdot g(11) + 3/4 \cdot g(10)$$
$$g(10.50) = 2/4 \cdot g(11) + 2/4 \cdot g(10)$$
$$g(10.75) = 3/4 \cdot g(11) + 1/4 \cdot g(10)$$
$$g(11) = 4/4 \cdot g(11) + 0/4 \cdot g(10)$$

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

```
% 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
```

```
    yellow=[1 1 0];  black=[0 0 0];
for y= __:__:__
    d= 1;
                  initializations necessary for each row
    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
                              y
end
```