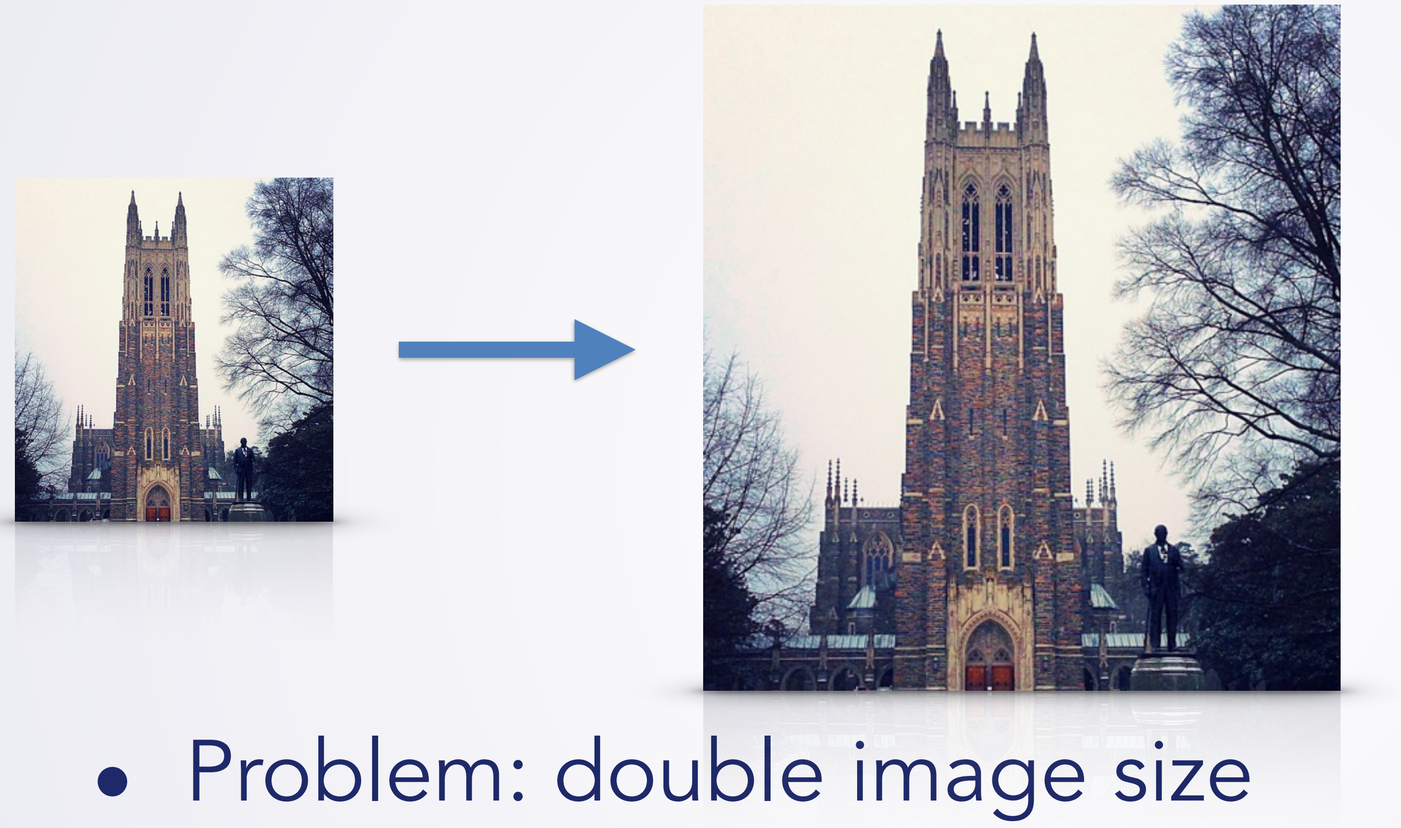
Enlarging an Image

Applying the Seven-step Process

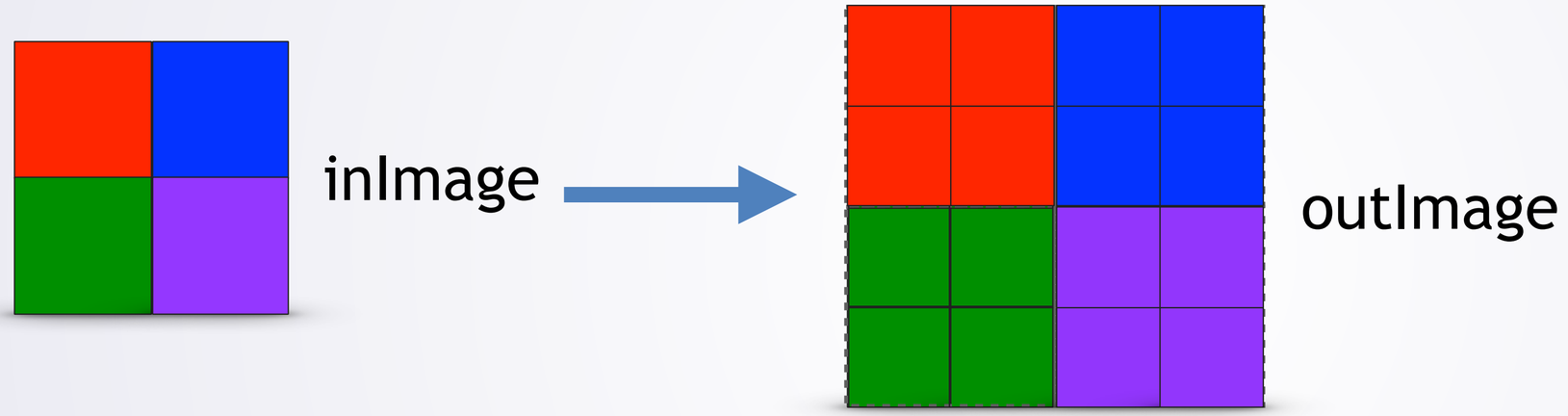
Duke
UNIVERSITY

Double Size of Image



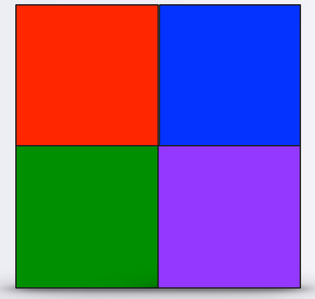
- Problem: double image size
  - Twice width + height
- 7 Steps

Step 1: Small Instance by Hand



- 2x2 → 4x4

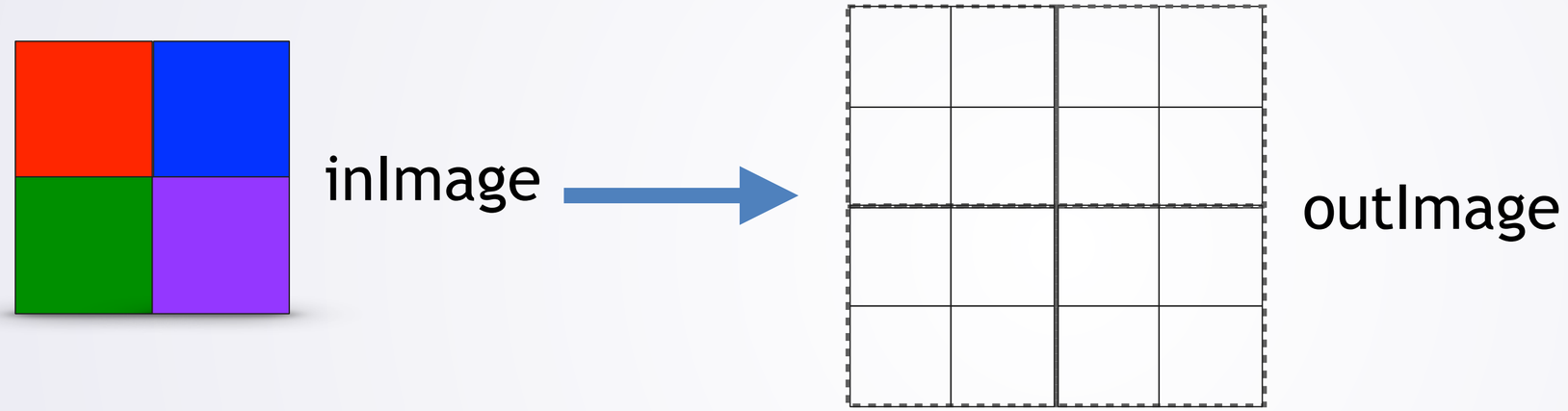
Step 2: Write Down Steps



inImage

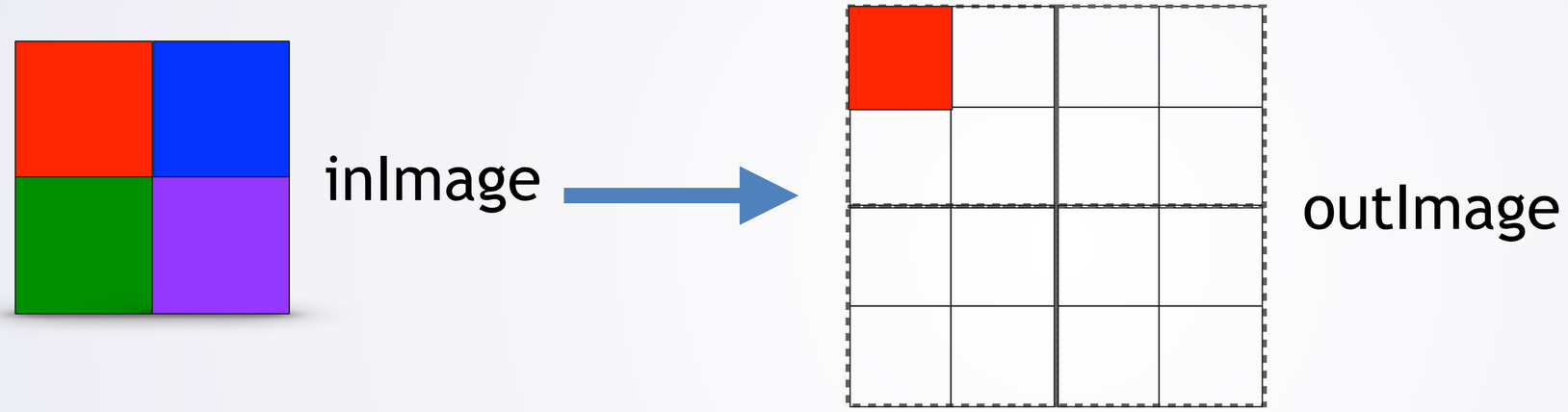
1. I started with the image I wanted (inImage)

Step 2: Write Down Steps



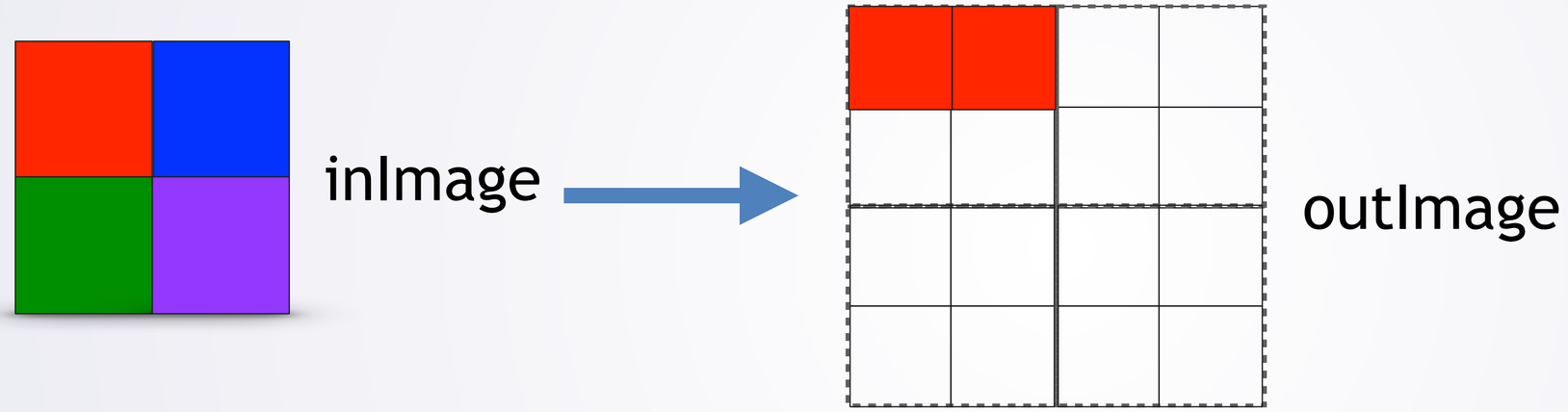
2 I made a blank image (outImage) twice as wide and twice as tall as inImage

Step 2: Write Down Steps



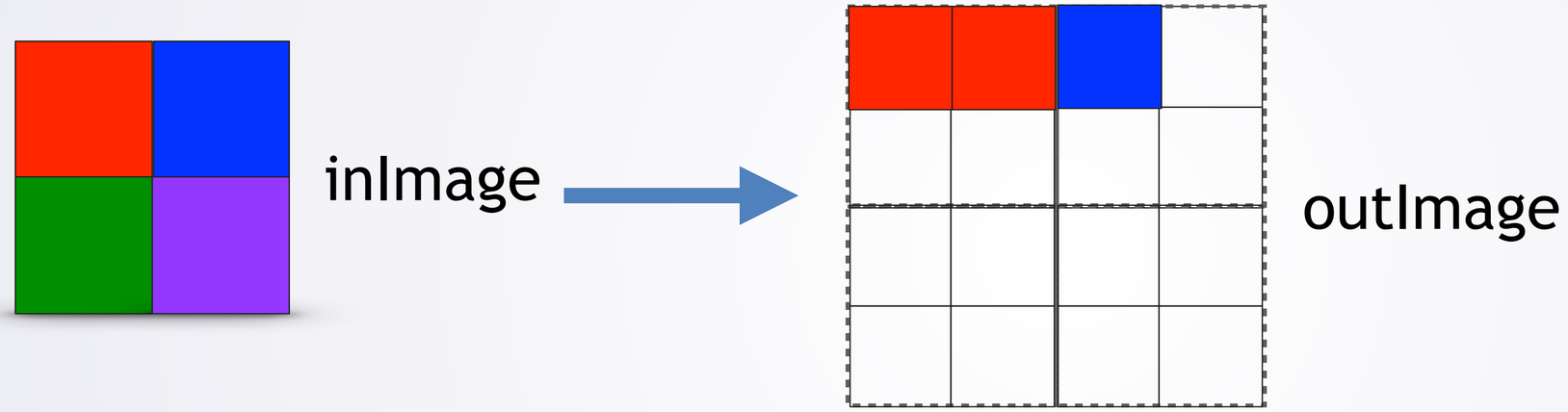
3. I made outImage's 1st pixel red

Step 2: Write Down Steps



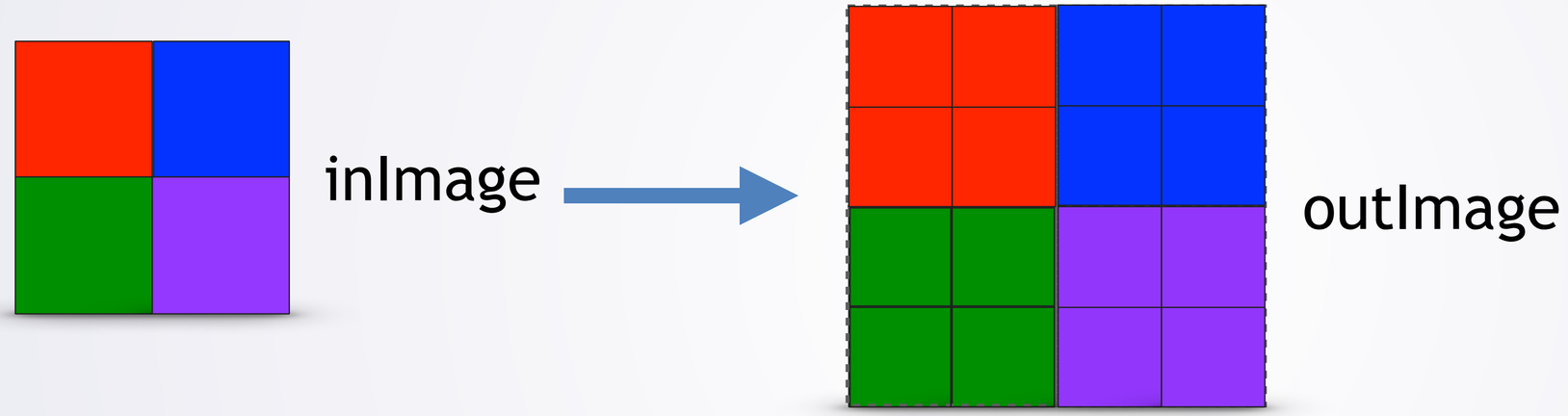
4 I made outImage's 2nd pixel red

Step 2: Write Down Steps



5 I made outImage's 3rd pixel blue

Step 2: Write Down Steps



18 I made outImage's 16th pixel purple

Step 2: Write Down Steps

1. I started with the image I wanted (inImage)
2. I made a blank image (outImage)
   twice as wide and twice as tall as inImage

3. I made outImage's 1st pixel red
4. I made outImage's 2nd pixel red
5. I made outImage's 3rd pixel blue
6. I made outImage's 4th pixel blue
7. I made outImage's 5th pixel red
8. I made outImage's 6th pixel red
9. I made outImage's 7th pixel blue
10. I made outImage's 8th pixel blue
11. … 9th pixel green
12. …10th pixel green
13. …11th pixel purple
14. …12th pixel purple
15. …13th pixel green
16. …14th pixel green
17. …15th pixel purple
18. …16th pixel purple

Step 3: Find Patterns

Look for repetitions and patterns

3. I made outImage's 1st pixel red
4. I made outImage's 2nd pixel red
5. I made outImage's 3rd pixel blue
6. I made outImage's 4th pixel blue
7. I made outImage's 5th pixel red
8. I made outImage's 6th pixel red
9. I made outImage's 7th pixel blue
10. I made outImage's 8th pixel blue
11. … 9th pixel green
12. …10th pixel green
13. …11th pixel purple
14. …12th pixel purple
15. …13th pixel green
16. …14th pixel green
17. …15th pixel purple
18. …16th pixel purple

Step 3: Find Patterns

Look for repetitions and patterns

3. I made outImage's 1st pixel red
4. I made outImage's 2nd pixel red
5. I made outImage's 3rd pixel blue
6. I made outImage's 4th pixel blue
7. I made outImage's 5th pixel red
8. I made outImage's 6th pixel red
9. I made outImage's 7th pixel blue
10. I made outImage's 8th pixel blue
11. … 9th pixel green
12. …10th pixel green
13. …11th pixel purple
14. …12th pixel purple
15. …13th pixel green
16. …14th pixel green
17. …15th pixel purple
18. …16th pixel purple

Step 3: Find Patterns

Look for repetitions and patterns

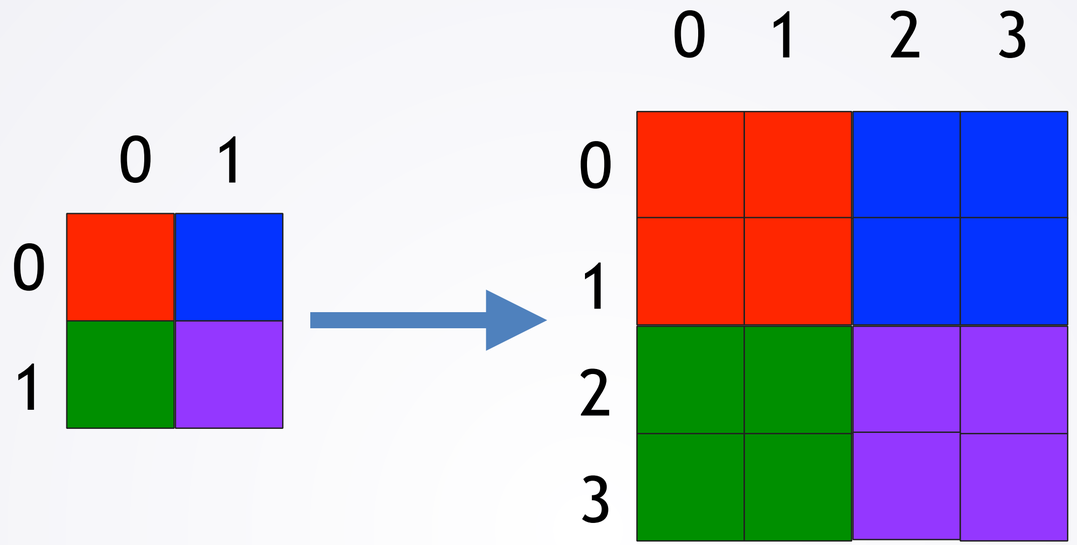
3. I made outImage's 1st pixel red
4. I made outImage's 2nd pixel red
5. I made outImage's 3rd pixel blue
6. I made outImage's 4th pixel blue
7. I made outImage's 5th pixel red
8. I made outImage's 6th pixel red
9. I made outImage's 7th pixel blue
10. I made outImage's 8th pixel blue
11. … 9th pixel green
12. …10th pixel green
13. …11th pixel purple
14. …12th pixel purple
15. …13th pixel green
16. …14th pixel green
17. …15th pixel purple
18. …16th pixel purple

Step 3: Find Patterns

Look for repetitions and patterns

3. I made outImage's 1st pixel red
4. I made outImage's 2nd pixel red
5. I made outImage's 3rd pixel blue
6. I made outImage's 4th pixel blue
7. I made outImage's 5th pixel red
8. I made outImage's 6th pixel red
9. I made outImage's 7th pixel blue
10. I made outImage's 8th pixel blue
11. … 9th pixel green
12. …10th pixel green
13. …11th pixel purple
14. …12th pixel purple
15. …13th pixel green
16. …14th pixel green
17. …15th pixel purple
18. …16th pixel purple

Step 3: Find Patterns



- Find pattern in pixel used

To color pixel with x = …	*Look at pixel with x = …*
0	***0***
1	***0***
2	***1***
3	***1***

Step 3: Find Patterns

To color pixel with x	*x/2*	*floor(x/2)*	*Look at pixel with x = …*
0	*0*	*0*	*0*
1	*0.5*	*0*	*0*
2	*1*	*1*	*1*
3	*1.5*	*1*	*1*

- Divide by 2 is close
  - But only want integer part
- Similar pattern for y's

Step 3: Find Patterns

1. I started with the image I wanted (inImage)

2. I made a blank image (outImage) twice as wide and twice as tall as inImage

3. For each pixel in outImage:
    a. Computed x=floor(pixel's x/2)
    b. Computed y=floor(pixel's y/2)
    c. Set pixel to the same color as the pixel at (x,y) in inImage

Step 4: Test Algorithm

1. I started with the image I wanted (inImage)
2. I made a blank image (outImage) twice as wide and twice as tall as inImage
3. For each pixel in outImage:
    a. Computed x=floor(pixel's x/2)
    b. Computed y=floor(pixel's y/2)
    c. Set to the same color as the pixel at (x,y) in inImage

Now it's your turn.
Think about if your answer is right

Yes. Answer is right!

Now let's write code