

# CSE 8A Lecture 7

- Reading for next class: 5.2.3-5.3
- PSA4: due Monday at midnight
  - Remember your (complete) comments including partner history
- Exam 2: Coming up this Friday

**CLICKERS OUT!**

# Reading Quiz #8

```
public Picture scaleUp( int numTimes )  
{  
    . . .  
}
```

1. If this is the method definition in `Picture.java`,  
**what is the method call** for a **Picture** object named `picObj` that  
**returns a Picture object**?

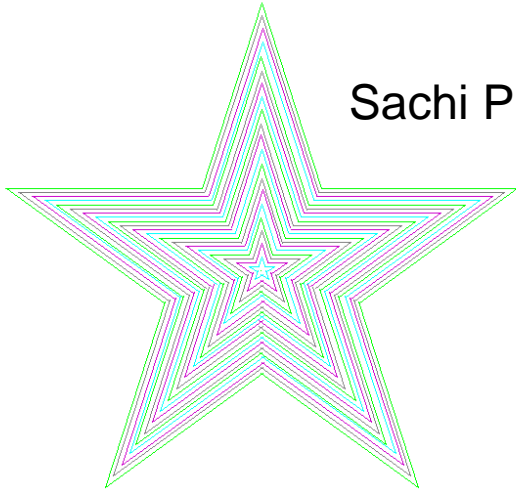
- A) `Picture picReturned = scaleUp(2);`
- B) `Picture picReturned = picObj.scaleUp(2);`
- C) `String picReturned = picObj.scaleUp(2);`
- D) `Picture = picObj.scaleUp(2);`

# Reading Quiz #8

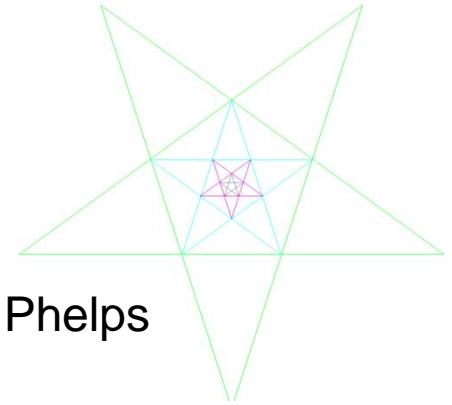
2. When can I define **two methods with the same name in the same class?**

- A) Never, that would cause a compiler error
- B) As long as one has a return statement and the other doesn't
- C) Only if the number or type of parameters are different
- D) Only if the parameters are the same

# PSA2 Gallery (Small, but nice)

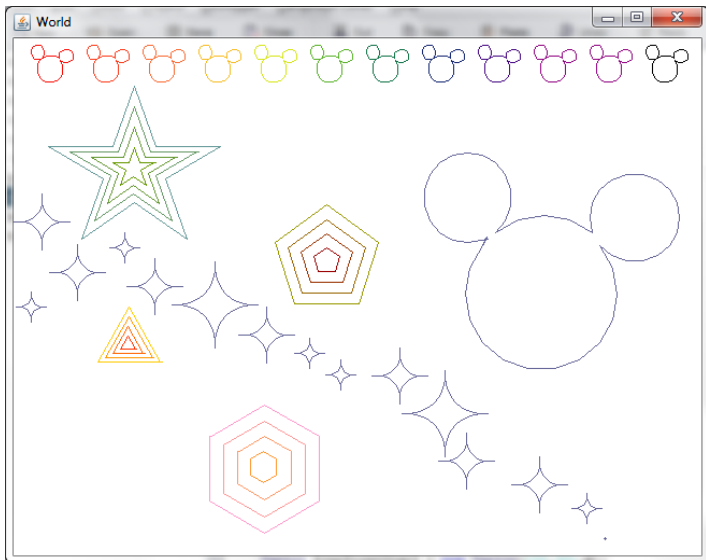


Sachi Pitkin and Sung Yoo

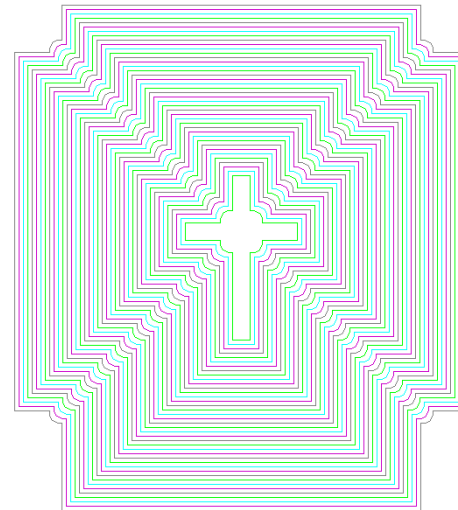


Tony Luo and Tony Phelps

Anonymous



Kevin Zhang and Subrahmanyam Parakala



40 turtles!!

# while

```
Pixel[] pixelArray = this.getPixels();
int index = 0;
while ( index < pixelArray.length )
{
    Pixel pix = pixelArray[index];
    pix.setGreen(255);
    index = index + 1;
}
```

# do...while

```
Pixel[] pixelArray = this.getPixels();
int index = 0;
do {
    Pixel pix = pixelArray[index];
    pix.setGreen(255);
    index = index + 1;
} while (index <= pixelArray.length);
```

# for

```
Pixel[] pixelArray = this.getPixels();
for ( int index = 0; index < pixelArray.length; index++ )
{
    Pixel pix = pixelArray[index];
    pix.setGreen(255);
}
```

# for each

```
Pixel[] pixelArray = this.getPixels();
for ( Pixel pix: pixelArray )
{
    pix.setGreen(255);
}
```

Which do you prefer?  
Why?

- 1) Solo: (30 sec)

2) Discuss: (2min)

3) Group: (30 sec)

# Nested Loops: How do they work?

## What order are pixels changed?

- A method in Picture.java... what does it print if width is 2 and height is 3?

```
Pixel p;  
for (int foo = 0; foo < getWidth(); foo++)  
{  
    for (int bar = 0; bar < getHeight(); bar++)  
    {  
        System.out.println( foo +" "+ bar );  
    }  
}
```

A 0 0  
0 1  
1 0  
1 1  
2 0  
2 1

B. 0 0  
1 0  
2 0  
0 1  
1 1  
2 1

C. 0 0  
0 1  
0 2  
1 0  
1 1  
1 2

D. 0 0  
1 1  
2 2

# Nested Loops: Tracing code

- A method in Picture.java... what does it print if width is 2 and height is 3?

```
Pixel p;  
for (int foo = 0; foo < getWidth(); foo++)  
{  
    for (int bar = 0; bar < getHeight(); bar++)  
    {  
        System.out.println(foo + " " + bar );  
    }  
}
```

foo

bar

- |                    |
|--------------------|
| 1) Solo: (30 sec)  |
| 2) Discuss: (2min) |
| 3) Group: (30 sec) |

# Nested Loops: How do they work? What order are pixels changed?

- A method in Picture.java...

```
Pixel p;  
for (int foo = 0; foo < getWidth(); foo++)  
{  
    for (int bar = 0; bar < getHeight(); bar++)  
    {  
        p = getPixel(foo, bar);  
        p.setColor(Color.BLACK);  
    }  
}
```



# What do these Picture methods do?

## What are their return types?

- `getPixel(int x, int y)`
- `getHeight()`
- `getWidth()`

- 1) Solo: (30 sec)  
2) Discuss: (2min)  
3) Group: (30 sec)

## Why does this have an error?

- In a method in Picture.java... assume height=50,width=100

```
Pixel p;  
for (int bar = 0; bar < getWidth(); bar++)  
{  
    for (int foo = 0; foo < getHeight(); foo++)  
    {  
        p = getPixel(foo, bar);  
        p.setColor(Color.BLACK);  
    }  
}
```

- A. It doesn't, this loops across rows, top to bottom
- B. It doesn't, this loops down columns, left to right
- C. It tries to index a pixel off the end of a row (x value too big)
- D. It tries to index a pixel off the end of a column (y value too big)

# Why did that have an error?

- The method `getPixel` in `Picture.java` with two parameters interprets the first one as an ‘x’ coordinate, and the second one as a ‘y’ coordinate of the Pixel to get
- When you call that method to get a Pixel from a Picture, it doesn’t matter what the names of the variables are *that you pass in!*
- `getPixel(foo,bar)` or `getPixel(bar,foo)` or `getPixel(x,y)` or `getPixel(y,x)`...
- The first parameter is always interpreted as the ‘x’ coordinate, and the second one as the ‘y’ coordinate, of the pixel you want

## How to fix that error

- Since **bar** takes values **0** to **getWidth()**, it is acting like an ‘x’ coordinate
- Since **foo** takes values **0** to **getHeight()**, it is acting like a ‘y’ coordinate
- So pass **bar** as first argument, and **foo** as second argument, to **getPixel**:

```
p = getPixel(bar, foo);
```

- (Better yet: write **x** instead of **bar** and **y** instead of **foo**; the computer doesn’t care, but it makes the code clearer to a human reader!)

# What's with foo and bar anyway?

The use of *foo* in hacker and eventually in programming context may have begun in MIT's Tech Model Railroad Club (TMRC)

*Foobar* may have derived from the military acronym FUBAR and gained popularity because it is pronounced the same.

–Wikipedia foobar page



Despite their popularity, **foo** and **bar** are NOT good choices for variable names

As the name of a bar, it's pretty good, though


# Some comments on style

```
Pixel p; for (int bar = 0; bar < getWidth(); bar++)  
{  
  for (int foo = 0; foo < getHeight(); foo++)  
    {p = getPixel(foo, bar);  
  p.setColor(Color.BLACK);}}
```

What's wrong with this code?

# Some comments on style

Meaningful variable names (generally more than 1 character)




```
Pixel pix;  
for (int xpos = 0; xpos < getWidth();  xpos++)  
{  
    for (int ypos = 0; ypos < getHeight(); ypos++)  
    {  
        pix = getPixel(xpos, ypos);  
        pix.setColor(Color.BLACK);  
    }  
}
```

One statement per line



Proper indentation (Dr. Java will help with this)



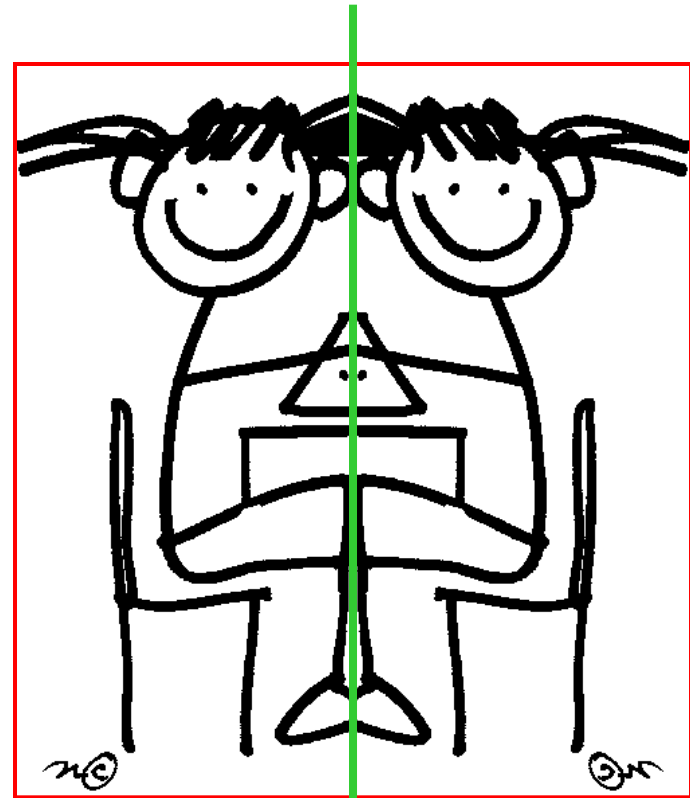
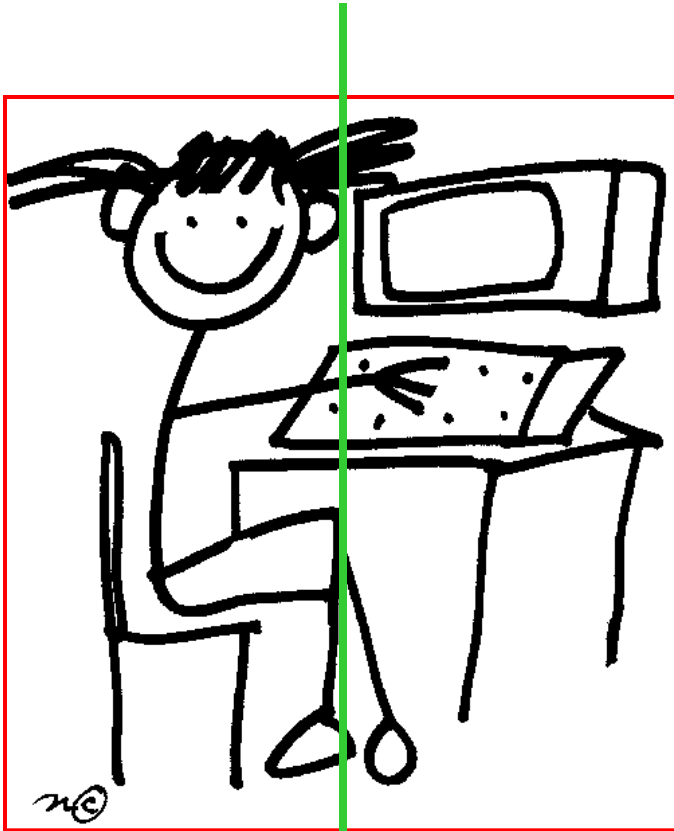
Lines not longer than  
80 characters





# Mirroring Around Vertical Axis

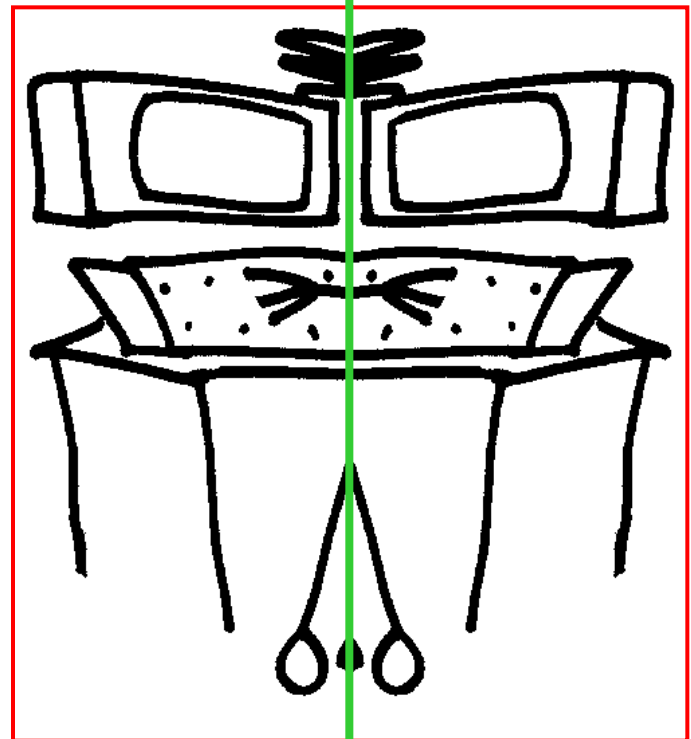
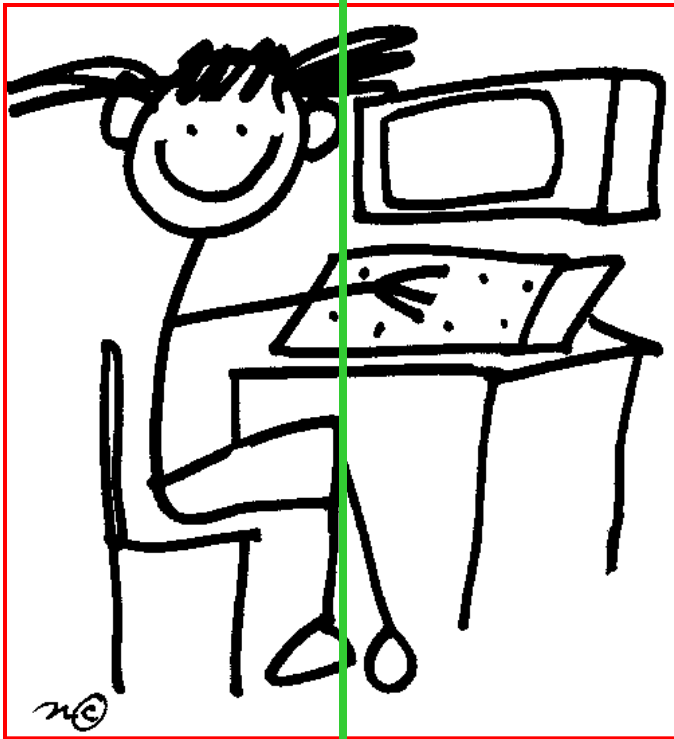
Mirror left to right



Vertical axis

# Mirroring Around Vertical Axis

Mirror right to left



Vertical axis

- |                    |
|--------------------|
| 1) Solo: (30 sec)  |
| 2) Discuss: (2min) |
| 3) Group: (30 sec) |

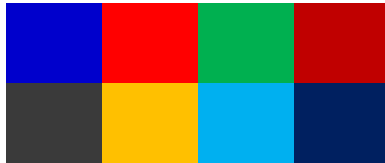
# Mirroring Around Vertical Axis: Left to Right

- What are the parameter values we use to index leftPixel and rightPixel for the first three iterations of the inner loop? (assume picture has a height = 50 and width = 100)

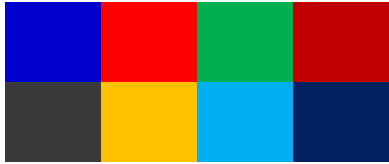
```
int mirrorPt = getWidth()/2;
Pixel leftP, rightP;
for (int y = 0; y < getHeight(); y++)
{
    for (int x = 0; x < mirrorPt; x++)
    {
        leftP = getPixel(x,y);
        rightP = getPixel(getWidth()-1-x,y);
        rightP.setColor(leftP.getColor());
    }
}
```

# How do you figure these kinds of questions out?

- Answer: Draw a diagram
  - imagine “beginning” and “answer”
  - Draw arrows to show how to get from beginning to answer
  - Then fill in numbers in order, write loops to create those numbers



# Mirroring Even Width versus Odd Width



```
int mirrorPt = getWidth() / 2;  
...  
for (int x = 0; x < mirrorPt; x++)
```

- 1) Solo: (30 sec)

2) Discuss: (2min)

3) Group: (30 sec)

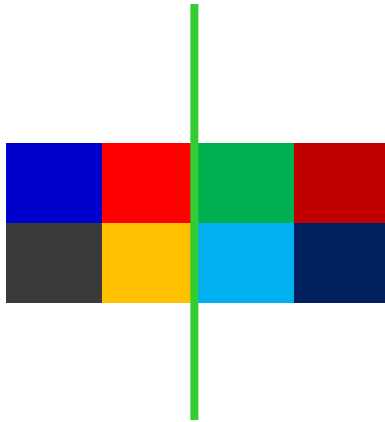
# Mirroring Odd-width Pictures

- What happens when this code attempts to mirror a Picture around the vertical axis when the Picture's width is odd (e.g. 101)?

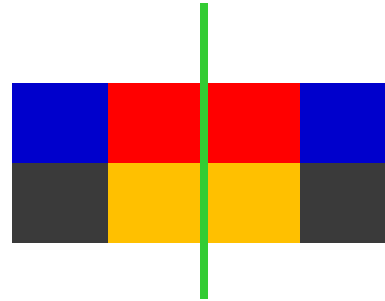
```
int mirrorPt = getWidth()/2;
Pixel leftP, rightP;
for (int y = 0; y < getHeight(); y++)
{
    for (int x = 0; x < mirrorPt; x++)
    {
        leftP = getPixel(x,y);
        rightP = getPixel(getWidth()-1-x,y);
        rightP.setColor(leftP.getColor());
    }
}
```

- A. It will work fine
- B. It will run, but it won't mirror correctly
- C. I won't run, there will be an index out of bounds exception
- D. It won't even compile if getWidth() is odd

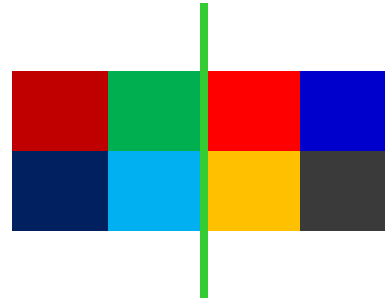
# Mirror versus “flip” (PSA4) (around vertical axis)



original



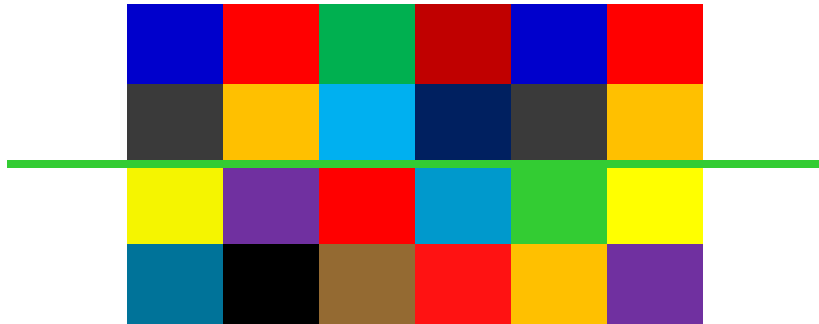
mirror



flip

- 1) Solo: (30 sec)
- 2) Discuss (1 min)
- 3) Group

What are the first (x,y) coords for  
topP and bottomP  
to mirror around **horizontal** axis?



topP      bottomP

A. (0,0)      (0,3)

(0,1)      (0,2)

(1,0)      (1,3)

B. (0,0)      (0,3)

(1,0)      (1,3)

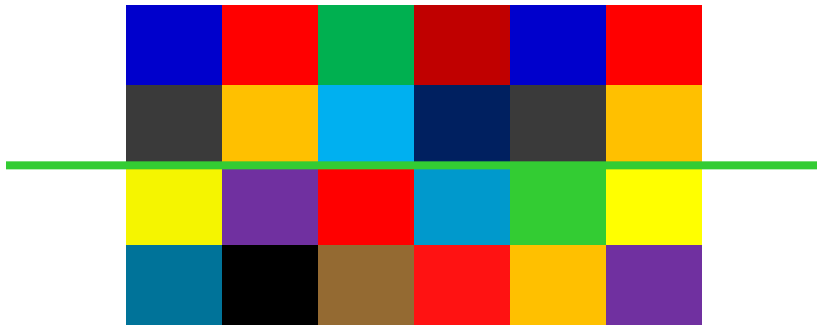
(2,0)      (2,3)

C. either A or B will work

D. none of the above



# Challenge: Complete the code that mirrors in the order specified by answer B



	topP	bottomP
B.	(0,0)	(0,3)
	(1,0)	(1,3)
	(2,0)	(2,3)

```
int height = getHeight();
```

```
int width = getWidth();
```

```
int mid = height/2;
```

```
Pixel topP, botP;
```

```
for (                                ) {  
    for (                            ) {  
        topP = getPixel(  
        botP = getPixel(  
        botP.setColor(topP.getColor());  
    }  
}
```

# TODO

- Study Exam#2 on Friday
- Reading for next class: 5.3.3-5.3.4
- Start PSA4 and show your cool images on Piazza!

