CSE 8A Lecture 10

- Reading for next class: 6.7-6.8
- Lab tomorrow! (check email for seating)
- PSA5 tonight + PSA6 tomorrow (discussion sections!)
- Use of “missed interview” is automatic
- NEW policy to earn “slip days” with partnerships
  - +0.5 every partnered PSA from now (psa5) on
  - -0.5 slip day from each partner when you need to turn in up to 24 h late (slip day is “split” among partners)
  - 24h late MAX (so 1 slip day only)
- Today:
  - NEW: If-statements
  - REVIEW: Variable scope, return value, comments
Interm Exam 2

• Nice job!
  – Median: 9
  – High score: 10 (105 students got it)

• Exams provide a minimum bar sanity check

• If you scored lower than a 7, it’s time to REALLY take action
  – Come to the lab/office hours and get help
  – Re-do all of the lab quizzes, clicker questions, reading questions, etc. and re-read all chapters so far.

CLICKERS OUT!
Exam2, Problem 1. While $\Rightarrow$ For. Parameter

// while loop version
public void takeSteps( int nSteps )
{
    int stepsTaken = 0;
    while ( stepsTaken < 5 ) {
        this.takeRandomStep( this.getRandomStep() ) ;
        stepsTaken = stepsTaken + 1;
    }
}

// POSSIBLE RESULT: for loop version
public void takeSteps( int nSteps )
{
    for (int stepsTaken = 0; stepsTaken < 5; stepsTaken++)
    {
        this.takeRandomStep( this.getRandomStep() ) ;
    }
}

public static void main(String[] args) {
    World w = new World();
    Turtle t = new Turtle( w );
    t.takeSteps( 10 );
}

turtle t is NOT doing 10 iterations. We need to use nSteps as loop condition (stepsTaken<nSteps)
Exam2, Problem 2. Nested Loops

// What does it print if width is 3 and height 3?

Pixel p;
for (int x = 0; x < getWidth(); x++)
{
    for (int y = 1; y < getHeight(); y++)
    {
        System.out.println(y + "#" + x);
    }
}

Values of x iterate from 0 to 2
Values y iterate from 1 to 2

1 # 0
2 # 0
1 # 1
1 # 2
2 # 1
2 # 2
Public void reducePartByHalfColor()
{
    Pixel[] pixArray = this.getPixels();
    Pixel p;
    int red, green, blue;
    for (int index=pixArray.length/4; index<pixArray.length/2; index++)
    {
        p = pixArray[index];
        red = p.getRed()/2;
        green = p.getGreen()/2;
        blue = p.getBlue()/2;
        p.setRed(red);
        p.setGreen(green);
        p.setBlue(blue);
    }
}
Chapter 6: Conditionally modifying pixels

All pixels change if COORDINATES meet criteria

All pixels change if COLOR meets criteria

All pixels change if meet both a COLOR and COORDINATE criteria
public void fillBottom(Color newColor)
{
    Pixel pix;
    for (int y = 0; y < this.getHeight(); y++)
    {
        for (int x = 0; x < this.getWidth(); x++)
        {
            <<<SELECT LINE OF CODE>>>>
            {
                pix = this.getPixel(x,y);
                pix.setColor(newColor);
            }
        }
    }
}

A) if(y<this.getHeight()/2)  B) if(y>this.getHeight()/2)
C) if (this.getPixel(x,y) < this.getHeight()/2)  D) if (this.getPixel(x,y) > this.getHeight()/2)
public void fillBottom(Color newColor) {
    Pixel pix;
    for (int y = 0; y < this.getHeight(); y++)
    {
        for (int x = 0; x < this.getWidth(); x++)
        {
            //<<<SELECT LINE OF CODE>>>>
            {
                pix = this.getPixel(x,y);
                pix.setColor(newColor);
            }
        }
    }
}

A. Yes, since we are changing the bottom half, we have to “fill in” across the rows in the inner loop

B. Yes, because we need to make sure the if statement is checking y not x

C. No, the if statement controls the assignment
How many times is the variable pix assigned a value?

A. 1
B. this.getWidth() times
C. this.getHeight() times
D. this.getHeight() * this.getWidth() times
E. this.getHeight() / 2 * this.getWidth() times

```java
public void fillBottom(Color newColor) {
    Pixel pix;
    for (int y = 0; y < this.getHeight(); y++) {
        for (int x = 0; x < this.getWidth(); x++) {
            pix = this.getPixel(x, y);
            pix.setColor(newColor);
        }
    }
}
```
public void everyOtherColumn(Color newColor)
{
    Pixel pix;
    for (int y = 0; y < this.getHeight(); y++)
    {
        for (int x = 0; x < this.getWidth(); x = x + 2)
        {
            pix = this.getPixel(x, y);
            pix.setColor(newColor);
        }
    }
}
How many iterations of the loop body are executed?

public void everyOtherColumn(Color newColor)
{
    Pixel pix;
    for (int y = 0; y < this.getHeight(); y++)
    {
        for (int x = 0; x < this.getWidth(); x = x + 2)
        {
            pix = this.getPixel(x,y);
            pix.setColor(newColor);
        }
    }
}

A. \((\text{getHeight()-1}) \times (\text{getWidth()-1})\)
B. \((\text{getHeight()-1}) \times (\text{getWidth()-1})/2\)
C. \(\text{getHeight()} \times \text{getWidth()}\)
D. \(\text{getHeight()} \times \text{getWidth()}/2\)
E. None of the above are always true
public void everyOtherColumn(Color newColor)
{
    Pixel pix;
    for (int y = 0; y < this.getHeight(); y++)
    {
        for (int x = 0; x < this.getWidth(); x++)
        {

            <<SELECT LINE OF CODE TO GO HERE>>
            pix = this.getPixel(x,y);
            pix.setColor(newColor);

        }
    }
}

A) if(x<this.getWidth()/2)  
B) if(x<this.getHeight()/2)
C) if ((x % 2) == 0)
D) if ( (this.getPixel(x,y) % 2) == 0)
If you can do it both ways, which is “better”? 

• Efficiency
  – How much time (computer instructions) does it take?
  – How much space (computer memory) does it take?

• Software Design
  – How readable is it for a human?
  – How easy is it to modify the code if needed, without introducing new bugs?
Which is more efficient?

• We just looked at these two approaches for setting pixel color in even-indexed columns:
  – (A) Looping over only some \((x,y)\) pixels
  – (B) Looping over all pixels, with if statement inside

• Which is more efficient?

• Why?
More complex control for color change

• What’s the most efficient way to reduce red eye?
  – A) Restricted loop bounds, only loop over x,y in range
  – B) Unrestricted loop bounds, use if statement to check for x,y in range

Pixels change if meet both a COLOR and COORDINATE criteria
public void removeRedEye (Color newColor, 
    int startx, int endx, int starty, int endy) {

    Pixel pix;
    for (int x =                 ; x <                ; x++)
    {
        for (int y =                ; y <                 ; y++)
        {
            pix = this.getPixel(x,y);
            if (pix.colorDistance(Color.red) < 167) 
                pix.setColor(newColor);
        }
    }
}
Let’s do it the “inefficient” way

• Assume we loop over ALL pixels in picture…

• What if statement would we use to check for pixels in range?
public void removeRedEye(Color newColor, int startx, int endx, int starty, int endy) {
    Pixel pix;
    for (int x = 0; x < getWidth(); x++)
    {
        for (int y = 0; y < getHeight(); y++)
        {
            if(
            
            )
            {
                pix = this.getPixel(x,y);
                if (pix.colorDistance(Color.red) < 167)
                    pix.setColor(newColor);
            }
        }
    }
}
Real-world example of if-else rules (among other things)

Eamonn Keogh, UCR

if (temp > 104.0)

ALARM!
Temp too high!
Real-world example of if-else rules (among other things)

TRILLIONS of data points (or more!)

How long would it take to loop through this data? How could you find patterns?

Eamonn Keogh, UCR
Usually, the sensors only display the last few minutes of data and figures such as the minimum and maximum temperature for that day. In most cases, the rest of the data is discarded. This is in part due to legal and privacy issues, which the researchers believe can be solved. It’s also because computer scientists didn’t have the tools to mine the vast amounts of data produced in pediatric intensive care units.

That changed after Keogh and a group of researchers recently developed a new technique, which allows for searching of datasets with more than one trillion objects. That’s a larger set than the combined size of all datasets in all data mining papers ever published.

http://ucrtoday.ucr.edu/9667
Keogh plans to use the archived data to develop algorithms that incorporate what he calls “if then rules” that can assist doctors. For example, if a heart beat looks like this, then a child may have difficulty breathing in five seconds.
Now quick review of a few concepts (you *probably* still need them in the next PSAs)

- Parameters
- Return values
- Variable scope
- Good commenting style
It’s nice to have code that is “user controllable”…

We have been hard-coding constants (40, 3, 100, for example) a lot, but we can write more flexible code using **PARAMETERS**

This lets us write code to do things like “cropping and pasting into a blank canvas”, but letting the user specify what part of the source picture to crop, and where to place it in the canvas.
public void copyKatiesXXX(

    Pixel sPixel, tPixel = null;
for (int sX = 40, tX = 100; sX < 110; sX++, tX++)
{
    for (int sY = 350, tY = 100; sY < 400; sY++, tY++)
    {
        sPixel = sourcePic.getPixel(sX,sY);
        tPixel = this.getPixel(tX,tY);
        tPixel.setColor(sPixel.getColor();
    }
}
}
Using parameters

In Picture.java...

```java
public void copyRegionTo (Picture target, int xSource,
        int ySource, int xTarget, int yTarget )
{
    Pixel sPixel, tPixel = null;
    for (int sX = xSource, tX = xTarget; sX < 100+xSource; sX++, tX++)
    {
        for (int sY = ySource, tY = yTarget; sY < 100+ySource; sY++, tY++)
        {
            sPixel = this.getPixel(sX,sY);
            tPixel = target.getPixel(tX,tY);
            tPixel.setColor(sPixel.getColor());
        }
    }
}
```

In main...

```java
Picture fish = new Picture( "fish.jpg" );
Picture blank = new Picture();
```

Write the code to copy the square at position (10, 50) in fish
to the blank canvas (vote on next slide)
Using parameters

In Picture.java…
public void copyRegionTo (Picture target, int xSource, int ySource, int xTarget, int yTarget )
{
   // Body omitted to save space
}

In main…
Picture fish = new Picture( "fish.jpg" );
Picture blank = new Picture();

A. fish.copyRegionTo(blank, 10, 50, 30, 30)
B. fish.copyRegionTo(blank)
C. fish.copyRegionTo()
In Picture.java...

```java
public void copyRegionTo (Picture target, int xSource, int ySource, int xTarget, int yTarget )
{
    // Body omitted to save space
}
```

In main...

```java
Picture fish = new Picture( "fish.jpg" );
Picture blank = new Picture();
fish.copyRegionTo(blank, 10, 50, 30, 30);
```

Variables only exist in the region they are defined. I.e. variables in main cannot be accessed in copyRegion and vice versa. The region where a variable exists is called its scope.
In Picture.java...

```java
public void copyRegionTo (Picture target, int xSource, int ySource, int xTarget, int yTarget )
{
    Pixel sPixel, tPixel = null;
    for (int sX = xSource, tX = xTarget; sX < 100+xSource; sX++, tX++)
    {
        for (int sY = ySource, tY = yTarget; sY < 100+ySource; sY++, tY++)
        {
            sPixel = this.getPixel(sX,sY);
            tPixel = target.getPixel(tX,tY);
            tPixel.setColor(sPixel.getColor());
        }
    }
}
```

In main...

```java
Picture fish = new Picture( "fish.jpg" );
Picture blank = new Picture();
fish.copyRegionTo(blank, 10, 50, 30, 30);
```
public void copyRegionTo (Picture target, int xSource, int ySource, int xTarget, int yTarget )
{
    Pixel sPixel, tPixel = null;
    for (int sX = xSource, tX = xTarget; sX < 100+xSource; sX++, tX++)
        for (int sY = ySource, tY = yTarget; sY < 100+ySource; sY++, tY++)
            sPixel = this.getPixel(sX,sY);
            tPixel = target.getPixel(tX,tY);
            tPixel.setColor(sPixel.getColor());
}

In main...
}
Parameters and scope

In Picture.java...

```java
public void copyRegionTo (Picture target, int xSource, int ySource, int xTarget, int yTarget )
{
    Pixel sPixel, tPixel = null;
    for (int sX = xSource, tX = xTarget; sX < 100+xSource; sX++, tX++)
    {
        for (int sY = ySource, tY = yTarget; sY < 100+ySource; sY++, tY++)
        {
            sPixel = this.getPixel(sX,sY);
            tPixel = target.getPixel(tX,tY);
            tPixel.setColor(sPixel.getColor());
        }
    }
}
```

In main...

```java
Picture fish = new Picture( "fish.jpg" );
Picture blank = new Picture();
fish.copyRegionTo(blank, 10, 50, 30, 30);
```
Parameters and return values

In Picture.java...

```java
public Picture copyRegionToNew(int xSource, int ySource, int xTarget, int yTarget )
{
    Picture newCanvas = new Picture();
    Pixel sPixel, tPixel = null;
    for (int sX = xSource, tX = xTarget; sX < 100+xSource; sX++, tX++)
    {
        for (int sY = ySource, tY = yTarget; sY < 100+ySource; sY++, tY++)
        {
            sPixel = this.getPixel(sX,sY);
            tPixel = newCanvas.getPixel(tX,tY);
            tPixel.setColor(sPixel.getColor());
        }
    }
}
```

In main...

```java
Picture fish = new Picture( "fish.jpg" );
Picture newCanvas = fish.copyRegionToNew(10, 30, 50, 50);
newCanvas.show();
```

What error will the following code produce?
A. This code will not compile
B. The line “Picture newCanvas = fish.copyRegionToNew…” in main will cause an error
C. The line newCanvas.show() will cause an error
In Picture.java...

```java
public Picture copyRegionToNew(int xSource, int ySource, int xTarget, int yTarget )
{
    Picture newCanvas = new Picture();
    Pixel sPixel, tPixel = null;
    for (int sX = xSource, tX = xTarget; sX < 100+xSource; sX++, tX++)
    {
        for (int sY = ySource, tY = yTarget; sY < 100+ySource; sY++, tY++)
        {
            sPixel = this.getPixel(sX,sY);
            tPixel = newCanvas.getPixel(tX,tY);
            tPixel.setColor(sPixel.getColor());
        }
    }
}
```

In main...

```java
Picture fish = new Picture( "fish.jpg" );
Picture newCanvas = fish.copyRegionToNew(10, 30, 50, 50);
newCanvas.show();
```

Some of copyRegionToNew’s variables

- `fish`
- `newCanvas`
- `this`
- `newCanvas`
Some comments on comments

/* A method to copy a 100x100 region of the calling object’s
* image to a blank canvas.
* xSource, ySource: the upper left corner of the
* region to be copied.
* xTarget, yTarget: the upper left corner where the region
* will appear in the new canvas.
* returns a new canvas with the region copied into it. */

public Picture copyRegionToNew(int xSource, int ySource, int xTarget, int yTarget) {
    Picture newCanvas = new Picture();
    Pixel sPixel, tPixel = null;
    for (int sX = xSource, tX = xTarget; sX < 100+xSource; sX++, tX++) {
        for (int sY = ySource, tY = yTarget; sY < 100+ySource; sY++, tY++) {
            sPixel = this.getPixel(sX, sY);
            tPixel = newCanvas.getPixel(tX, tY);
            tPixel.setColor(sPixel.getColor());
        }
    }
    return newCanvas;
}
TODO

• Reading for next class: 6.7-6.8
• Finish PSA5