CSE 8A Lecture 9

• Reading for next class: 6.2-6.3

• PSA5: Collage and Picture Flip, DON’T WAIT (it’s longer than the previous PSAs)

• Today (random useful stuff):
  – Variable scope
  – Returning a value
  – Comments
  – if statement
  – Review: loops
From the book: Cropping A Picture (page 147-148) – we’ll change it a bit

• Example of:
  – Working with both the calling object and a parameter object in a method
  • Extra information is passed to methods through parameters. The calling object is something like an extra parameter, named this

  – Doing something to a subset of the possible pixels in a picture
public void copyKatiesXXX( Picture sourcePic )
{
    Pixel sPixel = null, 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());
        }
    }
}

A. Feet
B. Part of dress
C. Hands
D. Part of Couch
E. Face
Parameters: getting information into methods

• It’s nice to have code that is “user controllable”…

• We have been hard-coding constants (e.g. 40, 3, 100) 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.
Underline values you would change into parameters and write a new method header

```java
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();
            }
}
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...

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...

```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();
```

A. `fish.copyRegionTo(blank, 10, 50, 30, 30);`
B. `fish.copyRegionTo(blank);`
C. `fish.copyRegionTo();`

Write the code to copy the square at position (10, 50) in fish to the blank canvas at position (30, 30)
Parameters and scope

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
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);
```
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 Picture.java...

In main...

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

Main's variables

fish blank
copyRegionTo's variables

target this
startxSource startySource
startxTarget startyTarget
Also sx, tx, sy, sy, sPixel, tPixel
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 copyRegionTo’s variables

- this
- newCanvas
/*
 * A method to copy a 100x100 region of the calling object’s
 * image to a blank canvas.
 *
 * xSource, ySource: upper left corner of region to be copied.
 * xTarget, yTarget: upper left corner where the region
 * will appear in 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;
}
while Loop Flow Chart

```java
int n = 1;

while ( n < 10 )
{
    System.out.println( n );
    n = n + 1;
}
```
```java
int n = 1;

do {
    System.out.println(n);
    n = n + 1;
} while (n < 3);
```
int n;
for( n = 1 ; n < 5 ; n++ )
    System.out.println( n );
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
if statement

```java
int n = 2;
if( n < 5 )
    n++;
System.out.println( n );
```
Select the if statement to make bottom half of picture some color.

```java
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 )
```java
int n = 2;
if( n > 5 )
    n = n + 1;
else
    n += 5;
System.out.println( n );
```
Challenge: Using methods (to do cool stuff)

public void copyRegionTo (Picture target, int xSource, int ySource, int xTarget, int yTarget)

Using method copyRegionTo (header above) write a method to copy a pattern of 20x20 squares from a source image to a target image (see below). Assume copyRegionTo has been modified to copy a 20x20 square instead of 100x100
TODO

• Reading for next class: 6.2-6.3
• PSA4 interview deadline Thursday noon