ANNOUNCEMENTS

MIDTERM

The Midterm will be Thursday, October 10, in RH-300 (Auditorium).
The midterm will be open notes.
Anything on paper is allowed
No electronic devices are allowed
  e.g. laptops, calculators, iPods, headphones
You must remain in the exam room until halfway through the exam
  Students who leave earlier than this will receive a zero on the exam
You must arrive no later than halfway through the exam
  Students arriving later than this will not be allowed to take the exam

MIDTERM

The midterm will cover everything from the class up to the exam. This includes:
  Lectures 1-10 (slides and material discussed in class)
  Labs, Assignments, & Exercises
In particular, students should focus on the following topics:
  Binary Numbers
    conversion from binary to decimal
  Java Primitive Types
  Variables
    declaration & initialization
    reference variables
  Arrays
  1 & 2 dimensional
  Methods
  Control Structures
    while
    for
    if
    if else
    if else else
  The best way to study is to be sure you fully understand all the examples from the lectures & all your lab assignments.
  Visit coaching hours to clarify any questions you have.
ICLICKER QUESTION

Which direction will the following code copy the array?

```java
for (x = 0; x < width/2; x++)
{
    for (y = 0; y < height; y++)
    {
        myArr[width-x-1][y] = myArr[x][y];
    }
}
```

A. Mirror top half to bottom  
B. Mirror bottom half to top  
C. Mirror left half to right  
D. Mirror right half to left

ICLICKER QUESTION

CONDITIONALS

REMOVE RED EYE

- Red eye is when the flash from the camera is reflected from the subject's eyes
- We want to change the red color in the eyes to another color
  - But not change the red of her dress
RED EYE ALGORITHM

• We can find the area around the eyes to limit where we change the colors
  – Using `pictureObj.explore()`
  – But we still just want to change the pixels that are “close to” red.
  – We can find the distance between the current color and our definition of red
    • And change the color of the current pixel only if the current color is within some distance to the desired color

DETAILED RED EYE ALGORITHM

• Pass the x and y of the starting location, and the x and y value of the end location.
• Using a nested loop through x and y:
  – Get the pixel at this x and y
  – Get the distance between the pixel color and red
  – If the distance is less than some value (167) then change the color to some passed new color

CONDITIONAL EXECUTION

• Sometimes we want a statement executed only if some expression is true
  – We can use the “if” statement in Java

```java
if (colorDistance < value)
    Statement or block to execute
next statement
```

USING IF EXERCISE

• Open DrJava and try this in the interactions pane

```java
int x = 2;
if (x > 1) System.out.println("X is > 1");
System.out.println("X is " + x);
x = 0;
if (x > 1) System.out.println("X is > 1");
System.out.println("X is " + x);
```
COLOR DISTANCE

• The distance between two points is computed as

\[ \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2} \]

(x1,y1) (x2,y2)

public void removeRedEye(int startX, int startY, int endX, int endY, Color newColor)
{
    Pixel pixelObj = null;

    // loop through the pixels in the rectangle defined by the // startX, startY, and endX and endY
    for(int x = startX; x < endX; x++)
    {
        for(int y = startY; y < endY; y++)
        {
            // get the current pixel
            pixelObj = this.getPixel(x,y);

            // if the color is near red then change it
            if(pixelObj.colorDistance(Color.RED) < 167)
                pixelObj.setColor(newColor);
        }
    }
}
**ICLICKER QUESTION**

What will be output by the following code?

```java
int x = 2;
if (x > 0)
    System.out.println("X is positive.");
System.out.println("X is " + x + ".");
```

A. X is positive. X is 2.
B. X is positive.
C. X is 2.
D. 2 is positive.
E. 2 is positive. 2 is 2.

---

**EDGE DETECTION**

- Find the areas of high contrast and turn pixels in this area black
  - Turn all other pixels white

---

**EDGE DETECTION ALGORITHM**

- To find areas of high contrast
  - Loop from y = 0 to y < height – 1
    - Loop from x = 0 to x < width
      - Get the pixel at the x and y (top pixel)
      - Get the pixel at the x and (y + 1) bottom pixel
      - Get the average of the top pixel color values
      - Get the average of the bottom pixel color values
      - If the absolute value of the difference between the averages is over a passed limit
        - Turn the pixel black
        - Otherwise turn the pixel white
USE IF AND ELSE FOR TWO POSSIBILITIES

- Sometimes you want to do one thing if the expression is true
- and a different thing if it is false

```java
int x = 200;
if (x < 128)
    System.out.println("<128");
else
    System.out.println(">=128");
```

EDGE DETECTION EXERCISE

- Write a method edgeDetection that takes an input limit
  - And turns all pixels black where the absolute value of the difference between that pixel and the pixel below is greater than the passed limit
  - And turns all pixels white where the absolute value of the difference between that pixel and the below pixel is less than or equal to the passed limit

```java
/**
 * Method to detect edges in a picture.
 * Color edges black and non-edges white.
 */
public void edgeDetection()
{
    Pixel topPixel = null;
    Pixel bottomPixel = null;
    for (int x = 0; x < this.getWidth(); x++)
    {
        for (int y = 0; y < this.getHeight()-1; y++)
        {
            topPixel = this.getPixel(x,y);
            bottomPixel = this.getPixel(x,y+1);

            if (topPixel.colorDistance(bottomPixel.getColor()) > 25)
                topPixel.setColor(Color.BLACK);
            else
                topPixel.setColor(Color.WHITE);
        }
    }
}
```

TESTING EDGE DETECTION

- Picture p = new Picture("../butterfly1.jpg");
- p.explore();
- p.edgeDetection(10);
- p.explore();
SEPIA-TONED PICTURES

- Have a yellowish tint, used to make things look old and “western.”

SEPIA-TONED ALGORITHM

- First make the picture grayscale.
- Change the shadows (darkest grays) to be even darker ($0 \leq \text{red} < 60$)
  - Decrease all the colors.
- Make the middle grays a brown color ($60 \leq \text{red} < 190$)
  - Decrease blue.
- Make the highlights (lightest grays) a bit yellow ($190 \leq \text{red}$)
  - Increase red and green
  - Or decrease blue

SEPIA-TONED ALGORITHM

- We need to be able to do some more things with our expressions before we can implement this.
- We have one condition that has two parts: $(60 \leq \text{red} < 190)$
  - This says “red is greater than or equal to 60 and red is less than 190.”
- How do we do this in Java?

CONDITIONAL OPERATORS

- We can check if several things are true - AND
  - Using && (evaluation stops if the first item is false)
  - Using & (to always evaluate both operands)
- We can check if at least one of several things is true - OR
  - Using || (evaluation stops if the first item is true)
  - Using | (to always evaluate both operands)
- We can check if only one of the things is true – Exclusive OR
  - Using ^
USING && (AND)

• Check that a value is in a range
  – For instance, is some value between 0 and 255 (inclusive) for valid pixel color values.
    \[ 0 \leq x \leq 255 \]
is written in Java as:
  \[ 0 \leq x \land x \leq 255 \]
  \[ x \geq 0 \land x \leq 255 \]
  \[ x \geq 0 \land 255 \geq x \]

USING || (OR)

• Check if at least one of several things is true
  – For instance, that at least one of red or green is 0.
  – In Java, this would be
    \[ \text{if}(\text{red} == 0 \lor \text{green} == 0 \lor \text{blue} == 0) \]
IClicker Question

What would the following test for?

if (red == 0 && green == 0 && blue == 0)

A. One of red, green, or blue is zero.
B. Gray.
C. White.
D. Black.
E. None of the above.

Truth Table

<table>
<thead>
<tr>
<th>Conditional</th>
<th>Operand 1</th>
<th>Operand 2</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>And</td>
<td>true</td>
<td>true</td>
<td>true</td>
</tr>
<tr>
<td>And</td>
<td>true</td>
<td>false</td>
<td>false</td>
</tr>
<tr>
<td>And</td>
<td>false</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>And</td>
<td>false</td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

Truth Table

<table>
<thead>
<tr>
<th>Conditional</th>
<th>Operand 1</th>
<th>Operand 2</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Or</td>
<td>true</td>
<td>true</td>
<td>true</td>
</tr>
<tr>
<td>Or</td>
<td>true</td>
<td>false</td>
<td>true</td>
</tr>
<tr>
<td>Or</td>
<td>false</td>
<td>true</td>
<td>true</td>
</tr>
<tr>
<td>Or</td>
<td>false</td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

Truth Table

<table>
<thead>
<tr>
<th>Conditional</th>
<th>Operand 1</th>
<th>Operand 2</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Exclusive Or</td>
<td>true</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>Exclusive Or</td>
<td>true</td>
<td>false</td>
<td>true</td>
</tr>
<tr>
<td>Exclusive Or</td>
<td>false</td>
<td>true</td>
<td>true</td>
</tr>
<tr>
<td>Exclusive Or</td>
<td>false</td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>
NOT CONDITIONAL OPERATOR

• Use ! To change the value to the opposite
  – !true is false
  – !false is true

NOT CONDITIONAL OPERATOR

• The ! operator can be used to change the meaning of expressions:
  !(x < 5) is "x is not less than 5"
  !(x == 10) is "x is not equal to 10"

NOT CONDITIONAL OPERATOR

• A not conditional operator applied to a complex conditional changes it:
  – !(op1 && op2) is equal to !op1 || !op2
  – !(op1 || op2) is equal to !op1 && !op2
• This is known as De Morgan’s Law

USING MULTIPLE IF STATEMENTS

• If we are doing different things based on a set of ranges
  if (0 <= x && x <= 5)
    Statement or block
  if (5 < x && x <= 10)
    Statement or block
  if (10 < x)
    Statement or block
USING MULTIPLE IF STATEMENTS

• If we are doing different things based on a set of ranges
  
  0 <= x < 6
  6 <= x < 7
  7 <= x < 8
  8 <= x < 9
  9 <= x

  if (0 <= x && x < 6)  
  Statement or block
  if (6 <= x && x < 7)  
  Statement or block
  if (7 <= x && x < 8)  
  Statement or block
  if (8 <= x && x < 9)  
  Statement or block
  if (9 <= x)  
  Statement or block

• If we are doing different things based on a set of ranges
  
  0 <= x < 6
  6 <= x < 7
  7 <= x < 8
  8 <= x < 9
  9 <= x

  if (0 <= x && x < 6)  
  System.out.println("Fail");
  else if (x < 7)  
  System.out.println("D");
  else if (x < 8)  
  System.out.println("C");
  else if (x < 9)  
  System.out.println("B");
  else  
  System.out.println("A");
**CONDITIONALS WITH > 2 CHOICES**

```java
public void sepiaTint()
{
    Pixel pixelObj = null;
    double redValue = 0;
    double greenValue = 0;
    double blueValue = 0;

    // first change the current picture to grayscale
    this.grayscale();

    // loop through the pixels
    for(int x = 0; x < this.getWidth(); x++)
    {
        for(int y = 0; y < this.getHeight(); y++)
        {
            // get the current pixel and color values
            pixelObj = this.getPixel(x,y);
            redValue = pixelObj.getRed();
            greenValue = pixelObj.getGreen();
            blueValue = pixelObj.getBlue();

            // tint the shadows darker
            if(redValue < 60)
            {
                redValue = redValue * 0.9;
                greenValue = greenValue * 0.9;
                blueValue = blueValue * 0.9;
            }

            // tint the midtones a light brown by reducing the blue
            else if(redValue < 190)
            {
                blueValue = blueValue * 0.8;
            }

            // tint the highlights a light yellow by reducing the blue
            else
            {
                blueValue = blueValue * 0.9;
            }
        }
    }
}
```
• `String file = FileChooser.getMediaPath("gorge.jpg");`
• `Picture p = new Picture(".../gorge.jpg");`
• `p.explore();`
• `p.sepiaTint();`
• `p.explore();`