ICLICKER QUESTION

What Java code when inserted into the \( \boxed{ } \) will cause the output to be 6?

A. index < 2
B. index < 3
C. index < 4
D. index < 5
E. index < 6

```java
int index = 1;
int count = 0;
while ( \boxed{index < 2} ) {
    count = count + index;
    index = index + 1;
}
System.out.println(count);
```
WHAT IS A TWO-DIMENSIONAL ARRAY?

- The pixels in a picture are really stored in a two-dimensional array.
  - Each pixel has an x value (horizontal location)
  - Each pixel has a y value (vertical location)
  - `pictureObj.getPixel(x,y)` returns the pixel at that location

EXAMPLE TWO-DIMENSIONAL ARRAYS

- Maps
  - That city is in zone D-5

EXAMPLE TWO-DIMENSIONAL ARRAYS

- Battleship game
  - Try I-5
    - Hit or miss
EXAMPLE TWO-DIMENSIONAL ARRAYS

- Chairs at a theater or game
  - Row C seat 20

ICLICKER QUESTION

A. [4,3]
B. [2,1]
C. [0,0]
D. [3,3]
E. All of the above are valid elements of the array

NESTED LOOP

- How would you get all the pixels in a picture using their x and y values?
  - One possibility is - from left to right and top to bottom?
NESTED LOOP

• How would you get all the pixels in a picture using their x and y values?
  – One possibility is - from left to right and top to bottom?
• How would you get all the pixels in a picture using their x and y values?
  – One possibility is - from left to right and top to bottom?

\[ \begin{array}{c}
\text{x} \\
\hline
\text{y} \\
\end{array} \]
• How would you get all the pixels in a picture using their x and y values?
  – From left to right and top to bottom?
  – [0,0] (i.e. x=0 and y=0), [1,0], [2,0], …
  – [0,1], [1,1], [2,1], …
  – [0,2], [1,2], [2,2], …
• We need to have one loop inside another
  – The outer loop counts y from 0 to height - 1
  – The inner loop counts x from 0 to width - 1

• To do this, we need to have two loops.
• Two nested loops: one loop inside another
  – The outer loop counts y from 0 to (height – 1)
  – The inner loop counts x from 0 to (width – 1)

• The outer loop makes sure we do each row
  – y = 0, then 1, 2, 3, … (height – 1)
• The inner loops makes sure that for each row, we do each element
  – x = 0, then 1, 2, 3, … (width-1)

// loop through the rows
for(int y = 0; y < this.getHeight(); y++)
{
  // loop through the columns
  for(int x = 0; x < this.getWidth(); x++)
  {
    // get the current pixel
    pixelObj = this.getPixel(x,y);
    // modify the color
    // put appropriate statements
    // set the new color
    pixelObj.setColor(colorObj);
  } // Inner for loop
} // Outer for loop

Sets which row we’re working on.
For the current row, do the element in each column.
THE OTHER WAY

- We could also do it in the other order:
  - Do it column by column
  - Within each column, make sure we do all the rows.

- This is equivalent to switching the two loops.
  - Making the x loop the outer loop.
  - Making the y loop in inner loop.

NESTED LOOP - THE OTHER WAY

```
// loop through the columns
for(int x = 0; x < this.getWidth(); x++)
{
  // loop through the rows
  for(int y = 0; y < this.getHeight(); y++)
  {
    // get the current pixel
    pixelObj = this.getPixel(x,y);
    // modify the color
    // put appropriate statements
    // set the new color
    pixelObj.setColor(colorObj);
  } // Inner for loop
} // Outer for loop
```

Sets which column we're working on. For the current row, do the element in each row.

ALTERNATIVE NESTED LOOP

- How would you get all the pixels in a picture using their x and y values
  - From top to bottom and left to right?

```
// loop through the columns
for(int x = 0; x < this.getWidth(); x++)
{
  // loop through the rows
  for(int y = 0; y < this.getHeight(); y++)
  {
    // get the current pixel
    pixelObj = this.getPixel(x,y);
    // modify the color
    // put appropriate statements
    // set the new color
    pixelObj.setColor(colorObj);
  } // Inner for loop
} // Outer for loop
```

USING 2D ARRAYS FOR PICTURES

- How can we use these 2D arrays to manipulate digital images?

- Let's try an example where we lighten all of the pixels in an image…
LIGHTEN THE COLOR ALGORITHM

• Start x at 0 and loop while x < the picture width (add 1 to x at the end of each loop)
  – Start y at 0 and loop while y < the picture height (add 1 to y at the end of each loop)
    • Get the pixel at this location
    • Get the color at the pixel
    • Lighten (brighten) the color
    • Set the color for the pixel to the lighter color

public void lighten()
{
  Pixel pixel = null;
  Color color = null;

  // loop through the columns (x direction)
  for(int x = 0; x < this.getWidth(); x++)
  {
    // loop through the rows (y direction)
    for(int y = 0; y < this.getHeight(); y++)
    {
      // get pixel at the x and y location
      pixel = this.getPixel(x,y);

      // get the current color
      color = pixel.getColor();

      // get a lighter color
      color = color.brighter();

      // set the pixel color to the lighter color
      pixel.setColor(color);
    }
  // Inner for loop
  }
  // Outer for loop
} // method lighten()

TRYING THE LIGHTEN METHOD

• In the interactions pane:
  String file = "c:/intro-prog-java/mediasources/caterpillar.jpg";
  Picture p1 = new Picture(file);
  p1.explore();
  p1.lighten();
  p1.explore();

CHANGING TO NESTED LOOP EXERCISE

• Change the method clearBlue() to use a nested for loop to loop through all the pixels
• Run the method again to check that it still works
• Check that the blue values are all 0 using pictureObj.explore()
VERTICAL MIRRORING

- What if we want to pretend to place a mirror in the middle of the picture?
  - We would see the left side of the picture mirrored on the right side.

MIRROR VERTICAL ALGORITHM

- How do we determine the x and y of the elements to copy from and to?
  - The source and destination.
- They are both in the same row, so they have the same y value.

MIRROR VERTICAL ALGORITHM

- The x values are harder.
- Consider the source first.
- We only want the elements in the left half.
- How do we express that?

- We know that each row has width.
  - That we can get from the getWidth() method.
- Half of that is width/2.
- Does that work?
- Two cases:
  - width is even.
  - width is odd.
MIRROR VERTICAL ALGORITHM

• The odd case:
  – What about width/2?
    • 3/2 = 1
    • 5/2 = 2
    • 7/2 = 3
  – So we will use elements from 0 to < width/2 as source.

MIRROR VERTICAL ALGORITHM

• The even case:
  – What about width/2?
    • 2/2 = 1
    • 4/2 = 2
    • 6/2 = 3
  – So we will use elements from 0 to < width/2 as source.

MIRROR VERTICAL ALGORITHM

• What about the destination?
  – For x = 0, we copy to (width-1).
  – For x = 1, we copy to (width-1)-1.
  – For x = 2, we copy to (width-1)-2.
• In general, copy x to (width-1)-x.

MIRROR VERTICAL ALGORITHM TO CODE

• We are going to need the midpoint
  int midpoint = this.getWidth() / 2;
• Loop through the rows (y values)
  for(int y = 0; y < this.getHeight(); y++) {
    – Loop through x values (up to midpoint)
      for(int x = 0; x < midpoint; x++) {
        • Set right pixel color to left pixel color
          Pixel leftPixel = this.getPixel(x, y);
          Pixel rightPixel = this.getPixel(this.getWidth() - 1 - x, y);
          rightPixel.setColor(leftPixel.getColor());
      }
  }

• The odd case:
  – What about width/2?
    • 3/2 = 1
    • 5/2 = 2
    • 7/2 = 3
  – So we will use elements from 0 to < width/2 as source.

MIRROR VERTICAL ALGORITHM

• The even case:
  – What about width/2?
    • 2/2 = 1
    • 4/2 = 2
    • 6/2 = 3
  – So we will use elements from 0 to < width/2 as source.

MIRROR VERTICAL ALGORITHM

• What about the destination?
  – For x = 0, we copy to (width-1).
  – For x = 1, we copy to (width-1)-1.
  – For x = 2, we copy to (width-1)-2.
• In general, copy x to (width-1)-x.
public void mirrorVertical()
{
    int mirrorPoint = this.getWidth() / 2;
    Pixel leftPixel = null;
    Pixel rightPixel = null;
    
    // loop through the rows
    for (int y = 0; y < this.getHeight(); y++)
    {
        // loop from 0 to just before the mirror point
        for (int x = 0; x < mirrorPoint; x++)
        {
            // set new right pixel color to left pixel color
            leftPixel = this.getPixel(x, y);
            rightPixel = this.getPixel(this.getWidth() - 1 - x, y);
            rightPixel.setColor(leftPixel.getColor());
        }
    }
}

TRYING MIRROR VERTICAL
• Create the picture
  - Picture pl
    = new Picture(FileChooser.getMediaPath("caterpillar.jpg");
• Invoke the method on the picture
  - pl.mirrorVertical();
• Show the picture
  - pl.show();

MIRROR HORIZONTAL
• What about mirroring around a mirror held horizontally in the vertical center of the picture?
  - Like a reflection in a lake?

MIRROR HORIZONTAL ALGORITHM
• Get the vertical midpoint
  - Picture height / 2
• Loop through all the x values
  - Loop from y=0 to y < vertical midpoint
    • Get the top pixel
      - At x and y
    • Get the bottom pixel
      - Height - 1 - y
    • Set the bottom pixel's color to the top pixel color
MIRROR HORIZONTAL EXERCISE

- Write the method to mirror the top half of the picture to the bottom half.
  - This is a motorcycle redMotorcycle.jpg
- How about mirroring bottom to top?
  - How would you write the method to do that?

ICLICKER QUESTION

Which of the two sets of loops corresponds to the order shown by the arrows?

A. for (int y = 0; y < 4; y++)
   { for (int x = 0; x < 4; x++)
     { // Access [x,y]
     }
   }

B. for (int x = 0; x < 4; x++)
   { for (int y = 0; y < 4; y++)
     { // Access [x,y]
     }
   }

C. Neither