Programming problem wrapup
What is the problem?

1: Must code a method that preprograms operations from G&E (book) Picture and other classes. You must know enough about the problem domain!

2: Method must be in the **RibbonablePicture extends Picture** class.

3: Must be named **diagAndVert**

4: Must paint the given figure.

5: Must have one int parameter variable.

6: Parameter must customize the lengths shown in the diagram.

7: Must know loops enough to combine **Pixel locating** with **Pixel getting** with **Pixel blackening** in loops to do the job.

(sideLen is the Picture's boundary. Do NOT program painting any of that.)
Problem Domain:
Digital Images
(There's a good introduction in G&Es writing)
Hidden, inside (you can't see them) one pixel are **THREE (3) (separate)** int variables

<table>
<thead>
<tr>
<th>Variable</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>redIntensity</td>
<td>0</td>
</tr>
<tr>
<td>greenIntensity</td>
<td>255</td>
</tr>
<tr>
<td>blueIntensity</td>
<td>0</td>
</tr>
</tbody>
</table>
In Labs, lects, labs, this Java code:

```java
Color purple = new Color( 175, 0, 175 );
Pixel pixRef;
pixRef = pRef.getPixel( 0, 0 );
pixRef.setColor( purple );
```

commands the computer to CHANGE the values of these 3 variables:

- redIntensity: 175
- greenIntensity: 255 0
- blueIntensity: 175
G&E's diagram of the same idea

This Pixel has almost max intensity in all 3 colors: It looks white.
This Pixel is LOCATED in location [0] of the 1-dimensional array of Pixels. It's LOCATED at x,y location (0,0) in the 2-dim. Picture (a matrix!)
To change the value in a single Java variable:
1. LOCATE the variable with its name.
   `length = 100;`
2. Do the copy ("assign") operation.

To change the values in the 3 color variables of a Pixel:
1. LOCATE and get a ref. to the Pixel with its x,y location.
   `pixRef = pRef.getPixel(5,0);`
2. Do (call) the `setColor` method.
   `pixRef.setColor(purple);`
How to use persons' API (Application Programming Interface)

- Read a book written by him and her about it.
- Read documentation that they wrote.
- Read sample applications they wrote.
- (Sometimes) read their source code
  - When the docs are deficient, including having bugs.
  - When you want to learn how it works or was structured.
  - When you want to debug or improve it.

Let's do it: Where are Pictures, Pixels and their methods defined?
What is the problem?

1: Must code a method that preprograms operations from G&E (book) Picture and other classes. You must know enough about the problem domain!

2: Method must be in the RibbonablePicture extends Picture class.

3: Must be named diagAndVert.

4: Must paint the given figure. CHANGE Pixel colors one by one.

5: Must have one int parameter variable.

6: Parameter must customize the lengths shown in the diagram.

7: Must know loops enough to combine Pixel locating with Pixel getting with Pixel blackening in loops to do the job.

Life is easy? You ONLY have to program painting of TWO skinny LINES of Pixels, not the whole Picture.
iClicker

Reminder: The width and height of this Picture are returned by the G&E book class method calls this.getWidth() and this.getHeight()

Question: How MANY Pixels should get their color changed to black?

(A) this.getWidth()
(B) this.getWidth() * this.getHeight()
(C) sideLen (parameter value)
(D) 2*sideLen-1 or 2*sideLen
(E) sideLen*sideLen
(A) \texttt{this.getWidth()}
how many Pixels in one whole row

(B) \texttt{this.getWidth() \ast this.getHeight()}
how many in the whole Picture

(C) \texttt{sideLen} (parameter value)
how many in just ONE of the TWO lines

(D) \(2 \ast \text{sideLen} - 1\) or \(2 \ast \text{sideLen}\)

(E) \texttt{sideLen \ast sideLen}
how many is a \texttt{sideLength \ast sideLength} square
Live Coding

• Set up the RibbonablePicture class
• Code the loops.
• Code a testing App.
• Make it more flexible by making the diagAndVert method take origin parameters.
• Make it more robust by conditionally modifying the Pixels to avoid painting outside the Picture