Car designers at General Motors Research Labs only sculpt half of a car out of clay and then use a vertical mirror to reflect that half to see the whole car. What if we want to see what a picture would look like if we placed a mirror on a vertical line in the center of the width of the picture to reflect the left side (Figure 7)?

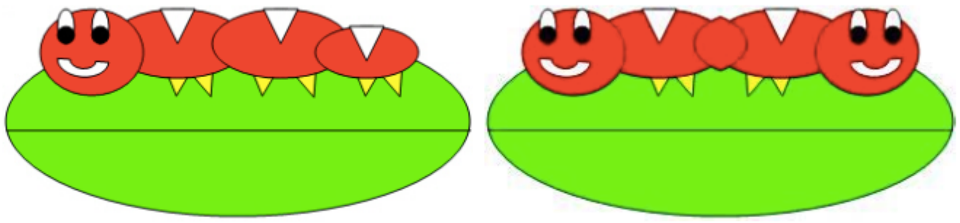
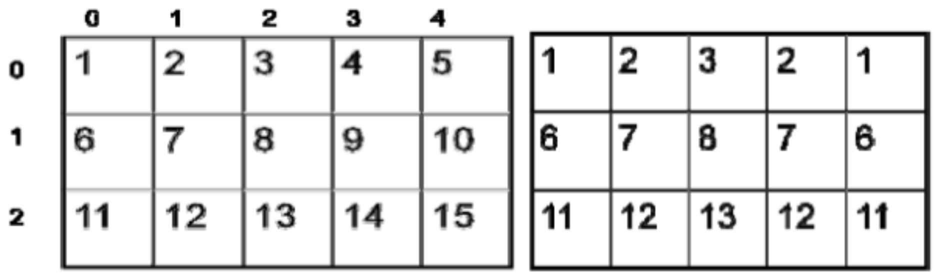


Figure 7: Original Picture (Left) and Mirrored Picture (Right)

How can we write a method to mirror a picture in this way? One way to figure out the algorithm, which is a description of the steps for solving a problem, is to try it on smaller and simpler data. Figure 8 shows the result of mirroring a two-dimensional array of numbers from left to right vertically. Note that the middle row is the "mirror line".



	0	1	2	3	4
0	1	2	3	4	5
1	6	7	8	9	10
2	11	12	13	14	15

1	2	3	2	1
6	7	8	7	6
11	12	13	12	11

Figure 8: Two-dimensional array of numbers (left) and mirrored result (right)

Exercises

1. Develop an algorithm for the mirroring process above. An algorithm does not need to be written in Java code. It can simply be a written explanation of a process. A cooking recipe is considered an algorithm. Test your algorithm on different sizes of two-dimensional arrays of integers. Will it work for 2D arrays that contain both even and odd number of columns?

One algorithm is to loop through all the rows and half the columns. You need to get a pixel from the left side of the picture and a pixel from the right side of the picture, which is the same distance from the right end as the left pixel is from the left end. Set the color of the right pixel to the color of the left pixel. The column number at the right end is the number of columns, also known as the width, minus one. So assuming there are at least 3 pixels in a row, the first left pixel will be at row=0, col=0 and the first right pixel will be at row=0, col=width-1. The second left pixel will be at row=0, col=1 and the corresponding right pixel will be at row=0, col=width-1-1. The third left pixel will be at row=0, col=2 and its right pixel will be at row=0, col=width-1-2. Each time the left pixel is at (current row value, current column value), the corresponding right pixel is at (current row value, width - 1 - (current column value)).

The method below implements the above algorithm.

```java
public void mirrorVertical()
{
  Pixel[][] pixels = this.getPixels2D();
  Pixel leftPixel = null;
  Pixel rightPixel = null;
  int width = pixels[0].length;
  for (int row = 0; row < pixels.length; row++)
  {
    for (int col = 0; col < width / 2; col++)
    {
      leftPixel = pixels[row][col];
      rightPixel = pixels[row][width - 1 - col];
      rightPixel.setColor(leftPixel.getColor());
    }
  }
}
```

You can test this with the testMirrorVertical method in the PictureTester class.

2. Write the method mirrorHorizontal that mirrors a picture around a mirror placed horizontally at the middle of the height of the picture. Mirror from top to bottom as shown in the pictures below. Write a class (static) test method in PictureTester to test this new method and call it in the main method.

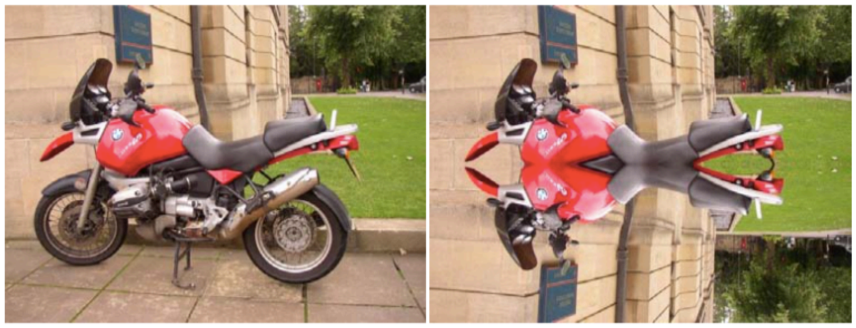


Figure 9: Original picture (left) and mirrored from top to bottom (right)