**HOLE FILLING:**

To describe a hole in an image, first I have colored a part of the image in white.  
The image is converted in grayscale.  
In each pixel which is colored in white I have assigned the value -1 at the appropriate index in the matrix.  
  
At each step of this process I have created an image that pops up.  
For example, the image “sunset.jpg” is loaded:   
1) A picture named “sunsetHole.jpg” with a hole is created   
2) A picture named “sunsetGrayScale.jpg” in grayscale is created  
3) A picture named “sunsetResult.jpg” with the hole filled is created

Note that if you want to change the size of the hole you need to change the indexes in the function makeTheHole in classImageConvertor.

1. For each pixel in the hole the algorithm iterates on all the pixels in the boundary, the cost of the operation inside the loop is O(1) so the complexity is O(n\*m).  
   Because a pixel can be 4 connected or 8 connected, we can say that m <= 4\*n or m <= 8\*n. The complexity of the algorithm is O(n^2).
2. We can approximate the result in O(n) using the following algorithm (Implemented in Class LinearHoleFiller):  
   For each pixel who is in the boundary, remove it from the boundary and look for his uncolored connected pixel.  
   Now, each uncolored pixel found is connected to colored pixels.  
   For each uncolored pixel found, color it with the average color of his connected colored pixel and add it to the boundary.  
   The algorithm iterates on all the pixels in the hole who is in O(n). Since the level of connectivity is 8 or 4, the operation in the loop is in O(1). The total time is in O(n).