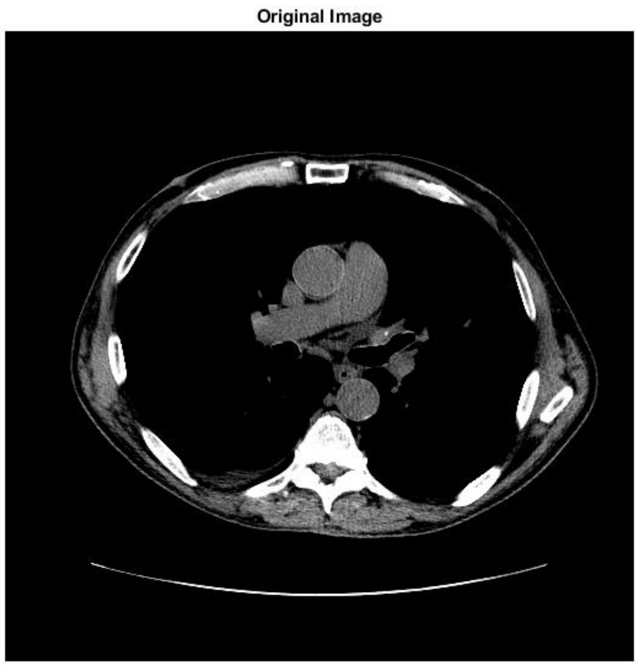
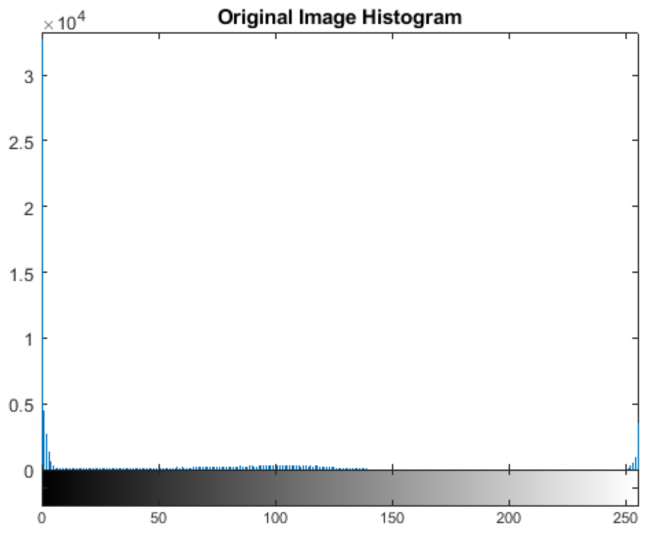
The method needs to start from a seed point that is determined by the user and then checks the eight neighbors to see if there is any point that can fall in the region by meeting a certain condition. To make the code clear, each part has been discussed separately, as follows:

First the gray level image is opened in MATLAB and the histogram of the image is visualized.





According to the histogram of the image, the user can decide on how to define the upper and lower threshold corresponding to the desired region. For our case, we are interested in taking the lungs region out from the CT image. this region falls within the darkest area with the range of [0 20]. The threshold has been defined by the difference between the upper and lower thresholds divided by 2.

Also, the user must define the seed point by choosing the desired point on the figure that pops up.

As an example, assuming the selected point is located in [248,158], and the values of upper and lower thresholds are selected as 20 and 0, respectively. So, the value of Thr would be 10.

Following the determined values, the region growing algorithm starts: First, we define some matrices with the size of the image to store the variables.

- *Seen_data: This matrix fills with ones in the location of each point that we are looking at. Using this matrix, we avoid looking at each pixel more than once during checking the neighbors of each seed.*

- *neighbour_label: This matrix fills with ones in the location of each neighbor that meets the condition. So, for each seed point, the neighbors that are below the threshold are labeled by 1.*
- *Output: This matrix stores the location of all neighbors of the seed points that lie in the region.*

Through the algorithm, the seed point is everchanging by turning each candidate in 'neighbour_label' as a new seed. Accordingly, the loop must continue as long as the seed matrix becomes empty or in other word, the 'length(Seed_x)' becomes zero.

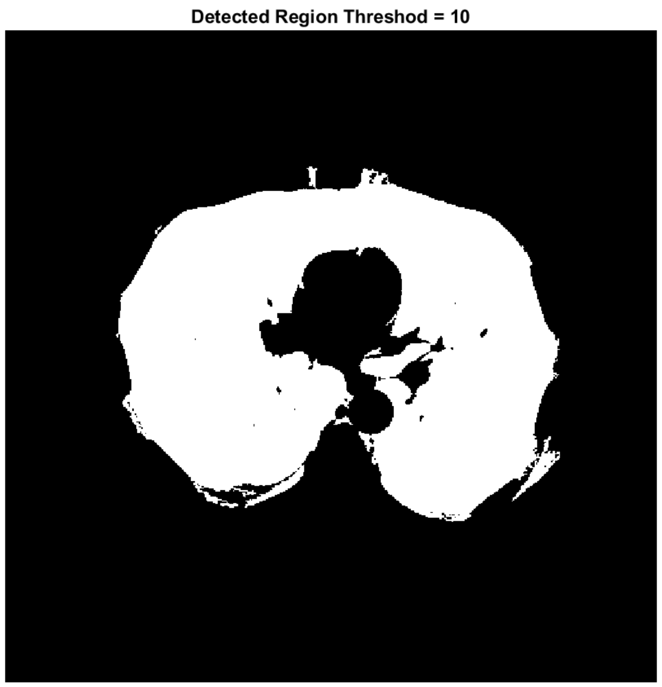
The 'for loop', is dedicated to the matrix of seed points in which the neighbors are evaluated. There are eight pixels around each seed for which the algorithm checks whether the difference between the corresponding unseen pixel and the seed falls below the threshold and if it is so, it labels the position of the pixel to 1, in the 'Seen_data' to ensure the pixel has already evaluated while the same location becomes 1 in 'neighbour_label', as well.

In the end, the 'Output' stores the values and position of 'neighbour_label'. The 'neighbour_label' becomes zero when the whole process of for loop is finished and all input seed points and the neighbors are evaluated. This means that the matrix may add 1, to the 'Output' more than once.

Once the for loop completes, the 'neighbour_label' is filled up with all the pixels in the neighborhood of each seed. These pixels are ready to check if they have the same condition in their adjacency.

In case there is no non-zero pixel in the 'neighbour_label', this means that the region has no more neighbor with the same condition and now we need to binarize the 'Output' so that we *can have mask of the selected region. So, all non-zero values in the 'Output' are found and become one. However, the matrix is a double image that becomes a logical image in the last step.*

The output of the algorithm is illustrated below:



The last step, is to show the region on top of the original image by another color. For this reason, we need to convert the gray level image into an RGB image by contaminating the gray levels.

Then, the locations of the non-zero values in the logical output are found and their intensities in each channel of the original image are replaced by the corresponding value in the desired color.

Masked region