Problem Statement 5:

Implement a Histogram equalization from scratch using C++ . Input should be an Image and the output should be a Linear filtered Image, Neat Documentation is expected with Code, Explanation, Input, and Output Image.

Solution:

Code:

```
#include <opencv2/opencv.hpp>
#include <iostream>

using namespace cv;
using namespace std;

int main(int argc, char** argv)
{
   // Read the image file
   Mat image = imread("C:\Users\neera\Pictures\Curceu");

   // Check for failure
   if (image.empty())
   {
      cout << "Could not open or find the image" << endl;
      cin.get(); //wait for any key press
      return -1;
   }

   //change the color image to grayscale image
   cvtColor(image, image, COLOR_BGR2GRAY);

   //equalize the histogram
   Mat hist_equalized_image;
   equalizeHist(image, hist_equalized_image);

   //Define names of windows
   String windowNameOfOriginalImage = "Original Image";
   String windowNameOfHistogramEqualized = "Histogram Equalized Image";

   // Create windows with the above names
   namedWindow(windowNameOfOriginalImage, WINDOW_NORMAL);
   namedWindow(windowNameOfHistogramEqualized, WINDOW_NORMAL);

   // Show images inside created windows.
```

```
    imshow(windowNameOfOriginalImage, image);
    imshow(windowNameOfHistogramEqualized, hist_equalized_image);

    waitKey(0); // Wait for any keystroke in one of the windows

    destroyAllWindows(); //Destroy all open windows

    return 0;
}
```

Explanation:

Code:

```
// Read the image file
Mat image = imread("D:/My OpenCV Website/fly-agaric.jpg");

// Check for failure
if (image.empty())
{
    cout << "Could not open or find the image" << endl;
    cin.get(); //wait for any key press
    return -1;
}
```

The above code segment will load the image from the specified file. The program will exit if the image load-up is failed.

```
//change the color image to grayscale image
cvtColor(image, image, COLOR_BGR2GRAY);
```

The above function converts the image in BGR (blue, green and red) color space to grayscale color space.

```
//equalize the histogram
Mat hist_equalized_image;
equalizeHist(image, hist_equalized_image);
```

The above function equalizes the histogram of the grayscale image and store the output in the hist_equalized_image.

```
//Define names of windows
String windowNameOfOriginalImage = "Original Image";
String windowNameOfHistogramEqualized = "Histogram Equalized Image";
```

```
// Create windows with the above names
namedWindow(windowNameOfOriginalImage, WINDOW_NORMAL);
namedWindow(windowNameOfHistogramEqualized, WINDOW_NORMAL);

// Show images inside the created windows.
imshow(windowNameOfOriginalImage, image);
imshow(windowNameOfHistogramEqualized, hist_equalized_image);
```

The above code segment will create windows and show images in them. As windows are created passing the flag WINDOW_NORMAL, they can be resized freely.

```
waitKey(0); // Wait for any keystroke in the window

destroyAllWindows(); //destroy all open windows

return 0;
```

The program will wait until any key is pressed. After a key is pressed, all created windows will be destroyed and the program will exit.

Output:

The above program used OpenCV library and run in Virtual Studio 2019.

Input Image:

Output Image: