Algorthims of filters

1-Black & White Filter Algorithm:

For each pixel (i, j) in the 'image' array

a- If the pixel value is greater than 127, set it to 255 (white)

b- Otherwise, set it to 0 (black)

2-Invert Filter Algorithm:

For each pixel (i, j) in the 'image' array

Set the pixel value to 255 minus its current value to invert the color

3-Merge Filter Algorithm:

1 -Prompt the user to enter the file name of the second image .

Add the ".bmp" extension to the image file name .

Read the second image into the 'image1' array .

For each pixel (i, j) in the 'image' array :

'a- Calculate the average of the pixel values at (i, j) in 'image' and 'image1

b- Set the pixel value in the 'image' array to the calculated average

4-Flip Image Algorithm:

1-Prompt the user to choose between horizontal (h) or vertical (v) flip .

If horizontal flip (h):

a- For each row (i) in the 'image' array

i- Swap the first half of the row with the second half

If vertical flip (v):

a- For each column (j) in the 'image' array-

i- Swap the upper half of the column with the lower half

5 -Darken and Lighten Image Algorithm:

1 -Prompt the user to choose between darkening (d) or lightening (l) the image

2 -For each pixel (i, j) in the 'image' array

a- If the user chooses to lighten (l), calculate a new pixel value as (255 - current pixel value) / 2 + current pixel value

b- If the user chooses to darken (d), calculate a new pixel value as current pixel value / 2

6 -Rotate Image Algorithm :

1- Prompt the user to choose between rotating the image by 90, 180, or 270 degrees

2- 'Create a new array 'rotatedImage' of the same size as 'image .

3-For each pixel (i, j) in the 'image' array :

'a- Based on the chosen angle, calculate the new position (newI, newJ) in the 'rotatedImage

b- Copy the pixel value from 'image' at (i, j) to 'rotatedImage' at (newI, newJ)

4- Copy the 'rotatedImage' back to the 'image' array to complete the rotation .

7- Detect Image Edges Algorithm:

1- 'Create a new array 'image2' of the same size as 'image

2 -For each pixel (i, j) in the 'image' array (excluding borders)

a -If the pixel in 'image' is black (0) and at least one of its 8 neighbors in 'image' is white (255), set the (0) corresponding pixel in 'image2' to black

b- Otherwise, set the corresponding pixel in 'image2' to white .(255)

3- Copy the content of 'image2' back to the 'image' array to complete the edge detection

8- Enlarge Image Algorithm:

1 -Prompt the user to select a quarter (1, 2, 3, or 4) to enlarge

2- For each pixel (i, j) in the 'image' array:

'a- Based on the selected quarter, calculate the corresponding position in 'image1

'b- Copy the pixel value from 'image1' at that position to 'image

9- Shrink Image Algorithm:

1- Prompt the user to choose the scaling factor (1/2, 1/3, or 1/4) .

2- Calculate the new width and height of the 'image' after scaling down .

3- Create a new array 'shrinkImg' with dimensions of the new width and height .

4- 'For each pixel in 'shrinkImg .4

a- Calculate the average color value of the corresponding block in the original 'image'

b- Set the pixel value in 'shrinkImg' to the calculated average

5- Copy the content of 'shrinkImg' back to the 'image' array to complete the shrinking .

6 -Fill the remaining border of the 'image' with white pixels .

10- :Mirror Image Algorithm:

1- Prompt the user to choose the mirror direction (l: left, r: right, u: upper, d: down) .

2- For each pixel (i, j) in the 'image' array:

a-Depending on the mirror direction chosen

If 'l' (left), copy the pixel value from 'image' at (i, j) to (i, SIZE - j - 1)

If 'r' (right), copy the pixel value from 'image' at (i, j) to (i, SIZE - j - 1)

If 'u' (upper), copy the pixel value from 'image' at (i, j) to (SIZE - i -1,j)

If 'd' (down), copy the pixel value from 'image' at (i, j) to (SIZE - i -1-j)

11- Shuffle Image Algorithm:

1-Prompt the user to enter a new order for the quarters of the image

2-'Create a new array 'image2' of the same size as 'image'

3-For each pixel (i, j) in the 'image' array:

a -Based on the specified order, copy the corresponding quarter from 'image' to 'image2'

4-Copy the content of 'image2' back to the 'image' array to complete the shuffle

12:-Blur Image Algorithm:

1-Create a new array 'image2' of the same size as 'image

2-For each pixel (i, j) in the 'image' array:

a- Calculate the average of the pixel values of the pixel and its 8 neighboring pixels-

b- Set the pixel value in 'image2' to the calculated average

3-Copy the content of 'image2' back to the 'image' array to complete the blur effect

13:- Crop Image Algorithm:

1.- Prompt the user to enter the coordinates (x, y), length (l), and width (w) for cropping .

2- .Create a new array 'croppedImage' with dimensions (l, w) .

3 -For each pixel (i, j) in the 'image' array, copy the corresponding pixel from 'image' to 'croppedImage' based on .
.the specified coordinates and dimensions

4- .Initialize the 'image' array with white pixels .

5 -Copy the content of 'croppedImage' back to the 'image' array to complete the croppin .

14 -Skew Right Algorithm:

1- .Prompt the user to enter the degree to skew the image to the right .

2. -Convert the degree to radians .

3. -Calculate the length (l) of the skewed image .3

. 4-Calculate the horizontal displacement (m) for skewing .

5.-Calculate the step size (s) for horizontal displacement .

6. -Calculate the pixel displacement per step (p) .

7.- 'Create a new array 'image2' of the same size as 'image .

8.- Initialize 'image2' with white pixels .

9: -For each pixel (i, j) in the 'image' array .

.a. Calculate the new horizontal position for the pixel based on the degree, m, and s

.b. Copy the pixel value from 'image' at (i, j) to 'image2' at the new position

10.-Copy the content of 'image2' back to the 'image' array to complete the right skew .

15- Skew Up Algorithm:

1. -Prompt the user to enter the degree to skew the image upwards .

2. -Convert the degree to radians .

3. -Calculate the length (l) of the skewed image .

.4 -Calculate the vertical displacement (m) for skewing .

5.- Calculate the step size (s) for vertical displacement .

6. -Calculate the pixel displacement per step (p) .

7.- 'Create a new array 'image2' of the same size as 'image .

8- .Initialize 'image2' with white pixels .

9-For each pixel (i, j) in the 'image' array .

.a. Calculate the new vertical position for the pixel based on the degree, m, and s

.b. Copy the pixel value from 'image' at (i, j) to 'image2' at the new position

10- .Copy the content of 'image2' back to the 'image' array to complete the upward skew .