Design Document Requirements

Name Isshin Furue   Date 4/28/2024

The overview of the project

The project allows users to load, display, edit, and save grayscale images represented by ASCII characters. The images consist of numerical values that represent different brightness levels, which are then translated into specific ASCII characters for display purposes.

The algorithm of the main function

```
Algorithm main()
    Initialize CurrentImage to an empty 2D array
    Initialize CurrentFilename to an empty string
    Display Main Menu
    While user does not choose to exit
        Read user's choice
        If choice is "Load Image"
            Call loadImage()
        Else if choice is "Display Image"
            Call displayImage()
        Else if choice is "Edit Image"
            Call editImage()
        Else if choice is "Save Image"
            Call saveImage()
        Else if choice is "Exit"
```

```
                Exit the program
End Algorithm
```

algorithms for each additional function

1. loadImage(filename: string):

```
Algorithm loadImage(filename)
    Try to open the file specified by filename
    If file is found
        Read data from file into CurrentImage
        Set CurrentFilename to filename
        Print "Image loaded successfully."
    Else
        Print "Error: File not found."
End Algorithm
```

2. displayImage(image: array):

```
Algorithm displayImage()
    If CurrentImage is not empty
        For each row in CurrentImage
            For each value in row
                Get corresponding ASCII character from BrightnessMapping
                Print character
            End For
            Print newline
        End For
    Else
        Print "No image loaded."
End Algorithm
```

3. editImage(image: array):

```
Algorithm editImage()
    Display Edit Menu (Crop, Dim, Brighten, Rotate)
    Read user's choice
    If choice is "Crop"
        Call cropImage()
    Else if choice is "Dim"
        Call dimImage()
    Else if choice is "Brighten"
        Call brightenImage()
    Else if choice is "Rotate"
        Call rotateImage90Degrees()
    Display the edited image by calling displayImage()
End Algorithm
```

4. saveImage(image: array, filename: string):

```
Algorithm saveImage(filename)
    Try to open/create file specified by filename
    For each row in CurrentImage
        For each pixel in row
            Write the corresponding ASCII character to file
        Write newline to file
    Close file
    Print "Image saved as filename."
End Algorithm
```

5. cropImage(image: array, startRow: int, startCol: int, numRows: int, numCols: int):

```
Algorithm cropImage(startRow, startCol, endRow, endCol)
    If specified coordinates are within the bounds of CurrentImage
```

```
        Create a new 2D array CroppedImage
        Copy values from CurrentImage within specified bounds to CroppedImage
        Set CurrentImage to CroppedImage
        Print "Image cropped successfully."
    Else
        Print "Invalid crop dimensions."
End Algorithm
```

6. dimImage(image: array):

```
Algorithm dimImage()
    For each pixel in CurrentImage
        If brightness level > 0
            Decrease the brightness level by 1
    Print "Image dimmed."
End Algorithm
```

7. brightenImage(image: array):

```
Algorithm brightenImage()
    For each pixel in CurrentImage
        If brightness level < maximum brightness level (4)
            Increase the brightness level by 1
    Print "Image brightened."
End Algorithm
```

8. rotateImage90Degree:

```
Algorithm rotateImage90Degrees()
```

```
    Create a new 2D array RotatedImage with swapped dimensions of CurrentImage
    For each pixel (i, j) in CurrentImage
        Set RotatedImage[j, length of CurrentImage - 1 - i] to CurrentImage[i, j]
    Set CurrentImage to RotatedImage
    Print "Image rotated 90 degrees."
End Algorithm
```