Programming Assignment Report: Image Reduction and Merge with Frame

Introduction

This report details the development process of a C program designed to perform image reduction by a factor of two and then merge four copies of the reduced image with a colored frame around the merged images. The program follows several steps, from reading a bitmap image file to applying transformations and outputting the final image with a frame.

Objective

The main goal of this assignment is to develop a C program capable of:

1. Reading a bitmap image from disk.
2. Reducing the size of the image by a factor of two.
3. Merging the reduced images into a single image with a specified frame size and color.
4. Writing the resultant image back to disk.

Methods

Image Reading and Header Extraction

The program begins by reading the input bitmap file and extracting its header information. This includes metadata such as image dimensions, bit depth, compression, and image size, crucial for understanding how to process the image data.

Image Reduction

The image is reduced by selecting pixels from even rows and columns. This operation effectively reduces the image's width and height by half, and the pixel data is stored for the merged image creation.

Merging and Framing

The reduced image is then copied into four quadrants. The first quadrant is a direct copy, the second is flipped vertically, the third is flipped horizontally, and the fourth is flipped both vertically and horizontally. A frame, whose width and color are specified by the user, is then added around the merged quadrants to create a single composite image.

Image Writing

Finally, the program writes the framed, merged image back to a new bitmap file. This includes writing the updated header and the modified image data.

Implementation Details

The implementation is divided into functions that handle specific tasks:

- `print_header()`: Prints the bitmap header information for debugging purposes.
- `write_image_file()`: Writes the image data and header to a new bitmap file.
- `main()`: Orchestrates the program's flow, calling the necessary functions and managing memory.

Each function is documented with comments explaining its purpose and the logic behind each significant line of code.

Results

The program successfully reduces the input image size by half, merges the reduced images as specified, and applies a frame with user-defined size and color. The final image is saved as a new bitmap file, demonstrating the program's ability to manipulate image data at the byte level.

Challenges

During development, challenges included handling the padding in bitmap rows to ensure alignment, correctly flipping the image data for mirroring, and ensuring that the framed image's dimensions were correctly calculated.

Conclusion

The completed C program fulfills the assignment requirements by applying image processing techniques at a low level, manipulating pixel data directly, and using the BMP file format's characteristics.