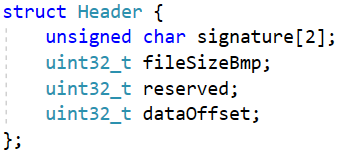
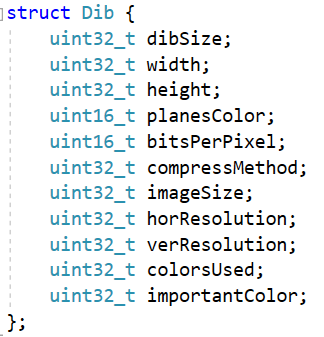
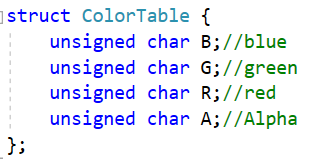
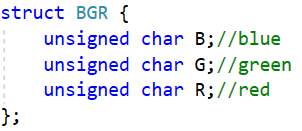
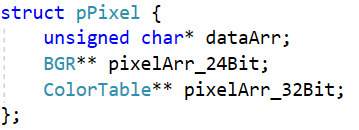
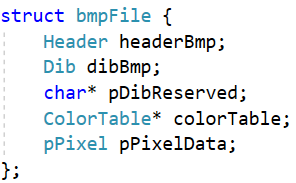
Tên: Nguyễn Thế Anh

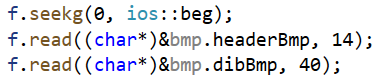
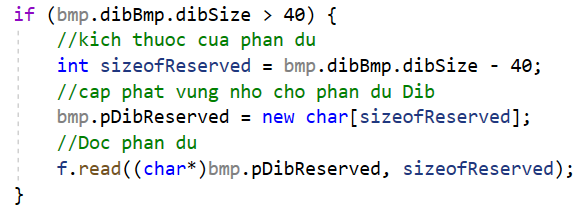
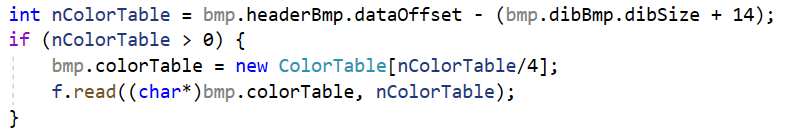
MSSV: 20120248

* **Cấu trúc dữ liệu**
* **#pragma pack(1)**

Với #pragma pack(1) giúp kích thước của struct được đảm bảo không có thêm phần đệm không mong muốn.

* **Header**
* Kích thước 14 bytes.
* Gồm:
* signature[2](2 bytes): Chữ ký file
* fileSizeBmp(4 bytes): Kích thước file ảnh bmp.
* reserved(4 bytes)
* dataOffset(4 bytes): Địa chỉ phần bắt đầu lưu dữ liệu điểm ảnh.
* Kiểu dữ liệu: uint32\_t: cố định kích cỡ 4 byte trên mọi nền tảng.
* **Dib**
* Kích thước 40 bytes.
* **ColorTable(4 bytes)**
* B(1 byte): blue
* G(1 byte): Green
* R(1 byte): Red
* A(1 byte): Alpha
* Dùng để lưu kiểu dữ liệu của pixel trong ảnh 32 bits và lưu bảng màu của ảnh 8 bits
* **BGR**
* B(1 byte): blue
* G(1 byte): Green
* R(1 byte): Red
* Dùng để lưu kiểu dữ liệu của pixel trong ảnh 24 bits
* **pPixel**
* unsigned char\* dataArr: mảng 1 chiều lưu dữ liệu điểm ảnh khi đọc file ban đầu(32, 24, 8 bits) và lưu dữ liệu điểm ảnh của ảnh 8 bits.
* BGR\*\* pixelArr\_24Bit: mảng 2 chiều kiểu BGR, lưu dữ liệu điểm ảnh của ảnh 24 bits được chuyển từ mảng dataArr sang.
* ColorTable\*\* pixelArr\_32Bit: mảng 2 chiều kiểu ColorTable, lưu dữ liệu điểm ảnh của ảnh 32 bits được chuyển từ mảng dataArr sang.
* **bmpFile**

hình ảnh

* Lưu tất cả dữ liệu của file ảnh
* Gồm: headerBmp, dibBmp, pDibReserved(phần dư của Dib nếu Dib Size > 40), colorTable(bảng màu), pPixelData(dữ liệu điểm ảnh).
* **Các hàm chính**
* **Hàm readBmpFile**
* Các tham số: char\* chỉ đến đường dẫn lưu ảnh, bmpFile& truyền tham chiếu do sau khi đọc thì dữ liệu trong bmpFile được cập nhật và thay đổi.
* Đọc header kích thước 14 bytes và Dib kích thước 40 bytes.
* Đọc phần dư (nếu có) 
* Đọc bảng màu(nếu là ảnh 8 bits, với kích cỡ cố định là 1024 bytes(256 màu \* 4 bytes)
* Đọc dữ liệu điểm ảnh vào mảng 1 chiều dataArr.
* Sau khi có dữ liệu dataArr, em lưu sang mảng 2 chiều pixelArr\_24Bit hoặc pixelArr\_32Bit bằng hàm getPixelArr.

Hàm getPixelArr:

Biến byteInRow: số bytes trên dòng của mảng pixel được tính bằng cách lấy số bytes trong mỗi pixel nhân với chiều rộng(width) rồi cộng với padding.

Cách lưu: pixelArr\_24Bit và pixelArr\_32Bit lưu các con trỏ trỏ đến địa chỉ đầu mỗi dòng. Sử dụng cách thức ép kiểu(BGR\* hoặc ColorTable\*) để padding nếu có cũng vẫn được tính là một phần tử. **Với cách lưu này thì dòng đầu tiên của pixelArr sẽ lưu các điểm ảnh nằm ở cuối.** Do các Pixel của ảnh bmp được lưu dưới dạng mảng 2 chiều với gốc (0;0) nằm ở góc trái dưới nên khi chuyển từ pixelArr sang mảng 1 chiều dataArr thì thứ tự ảnh sẽ đúng.

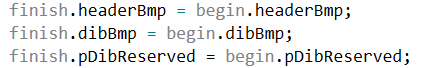
* **Hàm writeBmpFile**

Prototype function writeBmpFile

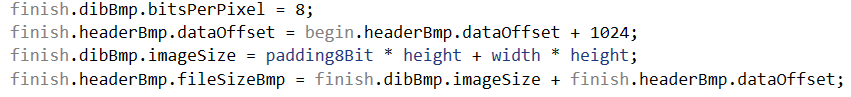
* Các tham số: char\*& chỉ đường dẫn cần ghi, bmpFile là ảnh được ghi vào đường dẫn
* Header và Dib được ghi tương tự như khi đọc file.
* Tiếp theo, ghi phần dư (nếu có)
* Tiếp tục ghi bảng màu (dấu hiệu: nếu dataOffset > (size Dib + Header))
* Cuối cùng là ghi dữ liệu điểm ảnh thông qua mảng 1 chiều dataArr. Mảng 1 chiều dataArr
* **Hàm convertTo8Bit**

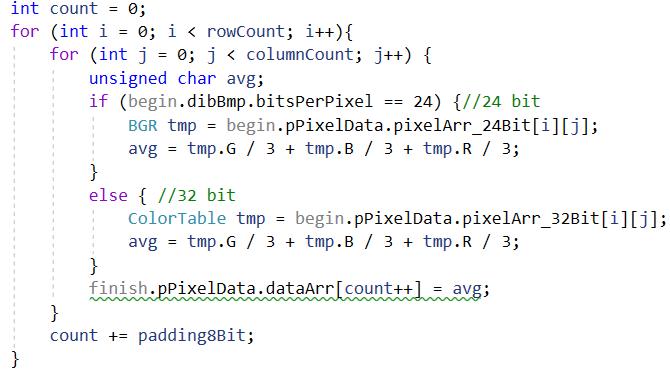
Prototype function convertTo8Bit

* Các tham số: begin là ảnh ban đầu, finish là ảnh dạng 8 bits của begin. Do sau khi hàm kết thúc, finish cập nhật dữ liệu mới nên được truyền tham số.
* Phương thức thực hiện:

Bước 1: Gán các thông tin của ảnh 24(hoặc 32 bits) sang ảnh 8 bits. Gồm header, dib và pDibReserved

Bước 2: Thay đổi các thông tin quan trọng. Gồm: Số byte trên mỗi pixel, padding, kích thước dữ liệu điểm ảnh và kích thước file

* Kích thước dữ liệu điểm ảnh = chiều dài \* chiều rộng + padding \* chiều dài.
* Kích thước file = kích thước dữ liệu điểm ảnh + lượng byte từ đầu đến điểm bắt đầu lưu dữ liệu điểm ảnh.

Bước 3: Cấp phát bộ nhớ cho mảng dataArr và thực hiện gán giá trị điểm ảnh. 

\*dataArr sẽ lấy giá trị trung bình của 3 điểm ảnh BGR (do ảnh 24 và 32 đều có các thông số BGR nên không cần chia trường hợp). Khi chạy hết 1 dòng sẽ được cộng thêm padding(của ảnh 8 bit mới) .

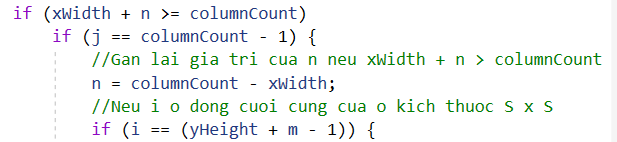
* **Hàm zoomOutPixel**

Các tham số: scr là ảnh cần thu nhỏ, dst là ảnh được thu nhỏ theo tỉ lệ S. dst sẽ được cập nhật giá trị nên được truyền tham chiếu.

Phương thức thực hiện:

Bước 1: Gán các thông tin header, dib, phần dư(nếu có), bảng màu(nếu có) cho ảnh dst(có chỉnh sửa width, height, file size, image size và padding) bằng hàm convertInforZoom.

Bước 2: [Trọng tâm] Gán giá trị cho mảng dataArr của dst.

* Cách thức: Chạy 2 vòng for(nhỏ). Mục đích là đi qua từng ô có kích thước S x S của ảnh scr. Sau đó dịch chuyển các mấu dòng 1 lượng S. Nếu hết dòng(dst) thì tăng mấu cột 1 lượng S và đưa mấu dòng về 0. Ràng buộc các điều kiện khi đến cuối dòng(scr) thì chỉ số(index) sẽ cộng thêm phần padding(nếu có). Vòng for dừng lại khi chỉ số(index) bằng với imageSize

Các vòng for

Điều kiện khi đến cuối dòng(dst)