BỘ GIÁO DỤC VÀ ĐÀO TẠO

TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN – ĐHQG TPHCM

~~~~~~*~~~~~~

# BÁO CÁO BÀI TẬP LỚN 1


| | | |
|---|---|---|
| *Sinh viên thực hiện* | : | **Đàm Tử Tâm** |
| *Mã số sinh viên* | : | **21120551** |
| *Môn học* | : | **Kỹ thuật lập trình** |
| *Giáo viên hướng dẫn* | : | **Phạm Minh Hoàng** |


**Thành phố Hồ Chí Minh – 2022**
~~~~~~

# I. Hàm main

**a. Mục đích:** gọi các hàm trong func.h và thực hiện chương trình

**b. Ý nghĩa các tham số:**

- argc: số lượng tham số truyền vào hàm main (theo yêu cầu đề bài tập)
- argv: các tham số truyền vào hàm, với argv[0] là tên chương trình

**c. Cách thiết kế hàm:**

- Khởi tạo các biến cần thiết để đưa vào hàm:
  - check: dùng để kiểm tra file ảnh input có hợp lệ hay không
  - header: cấu trúc dùng để lưu phần header của file ảnh input
  - dib: cấu trúc dùng để lưu phần dib của file ảnh input
  - newheader: cấu trúc dùng để lưu phần header của file ảnh output
  - newdib: cấu trúc dùng để lưu phần dib của file ảnh output
  - header8: cấu trúc dùng để lưu phần header của file ảnh 8 bit (dùng khi cần đọc ảnh 8 bit để lấy bảng màu)
  - dib8: cấu trúc dùng để lưu phần dib của file ảnh 8 bit (dùng khi cần đọc ảnh 8 bit để lấy bảng màu)
  - temp: dùng để lưu phần thừa của file ảnh nếu kích thước phần DIB>40
  - data: dùng để lưu dữ liệu điểm ảnh của file ảnh input
  - colortableA: dùng để lưu bảng màu của file ảnh 8 bit
  - newdata: dùng để lưu dữ liệu điểm ảnh của file ảnh output
  - temp8: dùng để lưu phần thừa của file 8 bit (dùng khi cần đọc ảnh 8 bit để lấy bảng màu)
  - data8: dùng để lưu dữ liệu điểm ảnh cho file 8 bit (dùng khi cần đọc ảnh 8 bit để lấy bảng màu)
  - file8: chuỗi lưu tên file ảnh 8 bit dùng để lấy bảng màu
- Kiểm tra nếu argc <4 hoặc >5 thì dừng chương trình (không đúng yêu cầu của chương trình)
- Chuyển ảnh sang trắng đen nếu argv[1] là -conv
  - Đọc ảnh input, nếu ảnh không hợp lệ thì giải phóng vùng nhớ rồi dừng chương trình
  - Nếu ảnh hợp lệ, tiếp tục chương trình: lấy bảng màu từ một ảnh 8 bit khác để gán sang ảnh output (lấy từ file ảnh "GetColorTable-8bit.bmp")
  - Gọi hàm chuyển ảnh sang trắng đen
  - Nếu ảnh input không hợp lệ (bpp không phải 24 hoặc 32) thì giải phóng vùng nhớ rồi dừng chương trình
  - Ghi ảnh output
- Thực hiện thu nhỏ ảnh theo tỉ lệ S nếu argv[1] là -conv

    + Nếu argc <5 thì dừng chương trình, ngược lại, ta tiếp tục thực hiện chương trình
    + Đọc ảnh input, nếu ảnh không hợp lệ thì giải phóng vùng nhớ rồi dừng chương trình
    + Nếu ảnh hợp lệ, tiếp tục gọi hàm thu nhỏ ảnh theo tỉ lệ S ở argv[4]
    + Ghi ảnh output
- Nếu argv[1] không phải "-zoom" hay "-conv" thì xem như tham số dòng lệnh không hợp lệ, dừng chương trình
- Thực hiện giải phóng vùng nhớ sau khi thực hiện chương trình (nếu chương trình là hợp lệ và vẫn chạy đến đây, nghĩa là chưa gặp bất cứ câu lệnh "return;" nào trong hàm main)

# II. Hàm readfile

**a. Mục đích:** dùng để đọc file ảnh bmp

**b. Ý nghĩa các tham số:**

- filepath: chuỗi kí tự là đường dẫn file ảnh input
- header: cấu trúc dùng để lưu phần header của ảnh input
- dib: cấu trúc dùng để lưu phần dib của ảnh input
- temp: con trỏ trỏ đến vùng nhớ dùng để lưu phần còn lại của dib nếu kích thước phần dib lớn hơn 40
- data: con trỏ trỏ đến vùng nhớ dùng để lưu dữ liệu điểm ảnh
- check: để kiểm tra ảnh hợp lệ hay không, nếu hợp lệ thì tiếp tục thực hiện các yêu cầu sau đó, còn không thì dừng chương trình
- colortableA: con trỏ trỏ đến vùng nhớ dùng để lưu phần bảng màu (nếu là ảnh 8 bit)

**c. Cách thiết kế hàm:**

- Mở file bmp và kiểm tra file bmp có mở được hay không
- Nếu file không mở được thì dừng chương trình, còn nếu mở được thì tiếp tục đọc phần header và kiểm tra signature của ảnh đã mở có phải là file ảnh bmp không. Nếu ảnh đã mở không phải file ảnh bmp thì dừng chương trình, ngược lại thì tiếp tục đọc phần dib
- Nếu kích thước phần dib lớn hơn 40 thì ta lưu phần còn lại vào temp
- Tiếp tục đọc bảng màu (chỉ đối với ảnh 8 bit)
- Tiếp tục đọc dữ liệu điểm ảnh
- Đóng file ảnh đã mở

# III. Hàm writefile

**a. Mục đích:** dùng để ghi file ảnh bmp

**b. Ý nghĩa các tham số:**

- filepath: chuỗi kí tự là đường dẫn file ảnh output
- header: cấu trúc lưu phần header của ảnh output
- dib: cấu trúc lưu phần dib của ảnh output
- temp: con trỏ trỏ đến vùng nhớ lưu phần còn lại của dib nếu kích thước phần dib lớn hơn 40
- data: con trỏ trỏ đến vùng nhớ lưu dữ liệu điểm ảnh
- colortableA: con trỏ trỏ đến vùng nhớ lưu phần bảng màu (nếu là ảnh 8 bit)

**c. Cách thiết kế hàm**

- Mở file để lưu ảnh và kiểm tra file có mở được hay không? Nếu mở được thì tiếp tục ghi file, ngược lại thì kết thúc hàm bằng câu lệnh return;
- Ghi phần header vào file ảnh
- Ghi phần dib vào file ảnh
- Nếu kích thước phần dib không phải 40 thì ta ghi phần dư vào file ảnh
- Nếu file ảnh cần lưu là ảnh 8 bit thì ta ghi bảng màu vào file ảnh
- Ghi phần dữ liệu điểm ảnh vào file ảnh

# IV. Hàm convert8bit

**a. Mục đích:** dùng để chuyển ảnh 24 bit hoặc ảnh 32 bit sang ảnh 8 bit (ảnh trắng đen)

**b. Ý nghĩa các tham số:** (ảnh input là ảnh 24 hoặc 32 bit, ảnh output là ảnh 8 bit)

- header: cấu trúc lưu phần header của ảnh input
- dib: cấu trúc lưu phần dib của ảnh input
- data: con trỏ trỏ đến vùng nhớ lưu dữ liệu điểm ảnh của ảnh input
- newheader: cấu trúc dùng để lưu phần header của ảnh output
- newdib: cấu trúc dùng để lưu phần dib của ảnh output
- newdata: con trỏ trỏ đến vùng nhớ dùng để lưu dữ liệu điểm ảnh của ảnh output
- check: để kiểm tra ảnh input hợp lệ hay không, nếu hợp lệ thì tiếp tục thực hiện các yêu cầu sau đó, còn không thì dừng chương trình

**c. Cách thiết kế hàm**

- Gán header và dib của ảnh input sang ảnh output
- Thay đổi giá trị của newdib.bpp (Bits per pixel) của ảnh output thành 8 và điều chỉnh lại offset (vì ảnh 8 bit có bảng màu)
- Tính số padding byte ảnh input và output, từ đó tính được kích thước dữ liệu ảnh và kích thước ảnh của ảnh output

- Khởi tạo mảng động chứa dữ liệu điểm ảnh của file output, kích thước mảng động được xác định dựa vào kích thước dữ liệu ảnh của ảnh output
- Tạo biến average dùng để tính trung bình cộng các giá trị BGR trong một pixel của ảnh input
- Tạo biến j để xác định vị trí trong mảng dữ liệu điểm ảnh của ảnh output, cùng với đó ta khởi tạo biến temp_j để xác định vị trí cần chèn padding byte trong mảng dữ liệu điểm ảnh của ảnh output
  - **Đối với ảnh input là ảnh 24 bit**
- Tạo vòng for để tính trung bình cộng của các giá trị Blue, Green, Red của từng pixel từ ảnh input rồi gán vào ảnh output
- Sau mỗi byte đã được tính ở phần dữ liệu điểm ảnh của ảnh output, ta tăng biến temp_j lên một đơn vị
- Khi temp_j = chiều ngang của ảnh output nghĩa là đã xử lý xong một dòng dữ liệu điểm ảnh, vì vậy sẽ chuyển sang chèn padding byte cho ảnh output rồi gán lại temp_j = 0 và tiếp tục xử lý dòng tiếp theo của dữ liệu điểm ảnh
  - **Đối với ảnh input là ảnh 32 bit**
- Tạo vòng for để tính trung bình cộng của các giá trị Blue, Green, Red của từng pixel từ ảnh input rồi gán vào ảnh output, đồng thời ở vòng for này ta tăng giá trị của biến i lên hai đơn vị thay vì một đơn vị để bỏ qua giá trị của Alpha
- Sau mỗi byte đã được tính ở phần dữ liệu điểm ảnh của ảnh output, ta tăng biến temp_j lên một đơn vị
- Khi temp_j = chiều ngang của ảnh output nghĩa là đã xử lý xong một dòng dữ liệu điểm ảnh, vì vậy sẽ chuyển sang chèn padding byte cho ảnh output rồi gán lại temp_j = 0 và tiếp tục xử lý dòng tiếp theo của dữ liệu điểm ảnh
  - **Ảnh input không phải là ảnh 24 bit hay 32 bit:**
- Xuất ra màn hình thông báo ảnh đầu vào không hợp lệ
- Gán check = NULL (ảnh input không hợp lệ) để không thực hiện các yêu cầu tiếp theo của chương trình
- Kết thúc hàm

# V. Hàm resize

a. **Mục đích:** thu nhỏ ảnh 8 bit, 24 bit hoặc 32 bit

b. **Ý nghĩa các tham số:** (ảnh input là ảnh đầu vào 8 bit, 24 bit hoặc 32 bit; ảnh output là ảnh 8 bit, 24 bit hoặc 32 bit tương ứng đã được thu nhỏ theo tỉ lệ S cho trước)

- header: cấu trúc lưu phần header của ảnh input
- dib: cấu trúc lưu phần dib của ảnh input

- newheader: cấu trúc dùng để lưu phần header của ảnh output
- newdib: cấu trúc dùng để lưu phần dib của ảnh output
- data: con trỏ trỏ đến vùng nhớ lưu dữ liệu điểm ảnh của ảnh input
- newdata: con trỏ trỏ đến vùng nhớ dùng để lưu dữ liệu điểm ảnh của ảnh output
- S: tỉ lệ cần thu nhỏ

**c. Cách thiết kế hàm**

- Gán header và dib của ảnh input sang ảnh output
- Khởi tạo biến x để lưu thêm 1 pixel cho chiều ngang ảnh output nếu chia dư chiều ngang ảnh input cho S có kết quả khác 0. Gán giá trị mặc định của x là 0
- Khởi tạo biến y để lưu thêm 1 pixel cho chiều dọc ảnh output nếu chia dư chiều dọc ảnh input cho S có kết quả khác 0. Gán giá trị mặc định của y là 0
- Khởi tạo biến oldpadding để tính số padding byte của ảnh input
- Khởi tạo biến temp_width để tính số pixel dư theo chiều ngang nếu chia chiều ngang ảnh cho S
- Khởi tạo biến temp_height để tính số pixel dư theo chiều dọc nếu chia chiều dọc ảnh cho S
- Tính chiều ngang ảnh output tạm thời vì chưa có padding byte và x
- Nếu temp_width khác 0, nghĩa là kết quả của chiều ngang ảnh input chia dư cho S khác 0, tăng chiều ngang của ảnh output lên 1 để lưu phần dư đó, gán x=1
- Tính chiều dọc ảnh output tạm thời vì chưa có padding byte và y
- Nếu temp_height khác 0, nghĩa là kết quả của chiều dọc ảnh input chia dư cho S khác 0, tăng chiều dọc của ảnh output lên 1 để lưu phần dư đó, gán y=1
- Biến temp_newimgsize để tính kích thước dữ liệu điểm ảnh output (chưa có padding byte)
- Biến padding để tính số padding byte của ảnh output
- Tính và gán lại kích thước dữ liệu điểm ảnh output chính thức (đã có padding byte), từ đó ta cũng tính và gán lại filesize ảnh output
- Cấp phát động cho newdata để lưu phần dữ liệu điểm ảnh của ảnh output
- Biến dem để xác định vị trí dữ liệu điểm ảnh của ảnh output trong mảng newdata
- Biến average để tính giá trị trung bình của các byte trong ô SxS
- Biến temp để xác định vị trí của các byte trong dữu liệu điểm ảnh input

- Biến temp_dem để xác định vị trí của temp là đang ở padding byte, từ đó ta sẽ chèn padding byte vào dữ liệu điểm ảnh của ảnh output
  - **Trường hợp thu nhỏ ảnh 8 bit**
- Tạo vòng for với biến i để đếm chiều dọc theo pixel của ảnh output (chưa tính y, biến i được khai báo ngoài vòng for)
  + Trong vòng for của i, tạo vòng với với biến j để đếm chiều ngang theo pixel của ảnh output (chưa tính x)
    - Tính tổng giá trị của SxS byte vào biến average rồi tính trung bình cộng giá trị của SxS byte dựa vào biến average và gán vào dữ liệu điểm ảnh của ảnh output. Sau đó, gán trở lại average = 0 để tiếp tục tính trung bình cộng byte tiếp theo
    - Tăng biến dem và temp_dem lên một đơn vị
    - Nếu temp_dem = newdib.width - x có nghĩa dem đã đi đến padding byte hoặc byte đã chia dư của ảnh output, ta đi vào điều kiện if:
      - Nếu x khác 0 nghĩa là có byte dư trên mỗi dòng (cột cuối trong ảnh input không có kích thước S*S), lúc này ta sẽ tính giá trị trung bình của các byte trên cột này theo kích thước S*temp_width lần lượt cho đến hết chiều cao được tính theo newdib.height - y và gán vào dữ liệu điểm ảnh của ảnh output
    - Đưa temp đến đầu ô S*S tiếp theo trong ảnh input
  + Tại vòng for của i, đưa temp đến dòng tiếp theo trong ảnh input cách trước đó một khoảng S dòng
- Khi đã tính và ghi hết dữ liệu điểm ảnh từ ảnh input vào ảnh output, nếu y khác 0 nghĩa là còn byte dư trong ảnh output (dòng cuối cùng trong ảnh input không có kích thước S*S), ta sẽ lần lượt tính trung bình cộng các byte của dòng này theo kích thước (newdib.width – x)*temp_height lần lượt cho đến hết theo chiều ngang newdib.width - x và gán vào kích thước dữ liệu điểm ảnh trong ảnh output. Lúc này ta đi vào điều kiện if:
  + Sau khi tính xong trung bình cộng các byte của dòng cuối cho đến hết theo chiều ngang newdib.width – x, ta xét điều kiện nếu x khác 0 nghĩa là có một cột không phải kích thước S*S, ta sẽ tính trung bình cộng các pixel cuối bên phải trong ảnh bitmap và gán vào dữ liệu điểm ảnh của ảnh output
  - **Trường hợp thu nhỏ ảnh 24 bit hoặc 32 bit**
- Khởi tạo biến dem_pixeldata dùng để đếm số byte đã đi qua và gán giá trị mặc định là 0
- Khởi tạo biến dem_newwidthvà gán giá trị mặc định là 0

- Khởi tạo biến next_pixel và gán giá trị mặc định là 1
- Tạo vòng for với biến i để đếm chiều dọc theo pixel của ảnh output (chưa tính y, biến i được khai báo ngoài vòng for)
    + Trong vòng for của i, tạo vòng với với biến j để đếm chiều ngang theo pixel của ảnh output (chưa tính x)
        - Tính tổng giá trị màu của B, G hoặc R của SxS byte vào biến average rồi tính trung bình cộng giá trị của SxS byte đó dựa vào biến average và gán vào một byte của dữ liệu điểm ảnh của ảnh output. Sau đó, gán trở lại average = 0 để tiếp tục tính trung bình cộng byte tiếp theo. Đồng thời tăng giá trị của dem_pixeldata lên một đơn vị
        - Tăng biến dem lên một đơn vị để giá trị điểm ảnh của ảnh output sẽ được gán vào byte tiếp theo ở lượt sau, đồng thời tăng biến temp_dem lên một đơn vị
        - Nếu dem_pixeldata = (newdib.bpp / 8) nghĩa là đã gán hết các giá trị ABRG hoặc BRG của một ô SxS của ảnh input vào một pixel của ảnh output, ta vào điều kiện if:
            - Tăng giá trị của dem_newwidth lên 1 đơn vị (dùng để tính vị trí tiếp theo của temp)
            - Gán lại giá trị của dem_pixeldata thành 0
        - Đưa temp đến đầu ô SxS tiếp theo
        - Nếu dem_pixeldata khác 0 nghĩa là vẫn chưa gán hết giá trị màu ABGR hoặc BGR của một ô SxS vào 1 pixel trong phần dữ liệu điểm ảnh của ảnh output, ta đưa temp đến byte chứa giá trị màu tiếp theo trong ảnh input. Ngược lại, nếu dem_pixeldata=0 nghĩa là ta đã gán hết giá trị màu ABGR hoặc BGR của một ô SxS vào 1 pixel trong phần dữ liệu điểm ảnh của ảnh output, ta gán lại next_pixel=1 để trong vòng lặp sau sẽ đưa temp đến đúng vị trị của byte màu tiếp theo trong ô SxS
        - Nếu temp_dem = ((newdib.width - x) * (newdib.bpp / 8)) nghĩa là temp đã gán hết các giá trị màu của ảnh input vào ảnh output (chưa tính x), ta đi vào điều kiện if:
            - Nếu x khác 0, ta gán tiếp các giá trị màu trong ô Sxtemp_width vào ảnh output
- Khi chương trình đã gán hết các giá trị màu của ảnh input vào ảnh output, trừ phần dư (newdib.width - x)*(dib.bpp / 8) x temp_height ở các dòng cuối trong phần dữ liệu điểm ảnh của ảnh input, nếu y khác 0 nghĩa là phần dư này tồn tại, ta tiếp tục gán phần dư này vào ảnh output

- Khi đã gán xong phần dư (newdib.width - x)*(dib.bpp / 8) x temp_height ở các dòng cuối trong phần dữ liệu điểm ảnh của ảnh input vào ảnh output, ta tiếp tục xét nếu x khác 0 nghĩa là còn ô cuối cùng trong giá trị điểm ảnh của ảnh input ta chưa gán vào ảnh output nên ta tiếp tục tính trung bình cộng các giá trị màu của ô này trong ảnh input rồi tiếp tục vào ảnh output