BMP 파일 구조의 이해

이종구

1. 목차

1. 과제 개요

2. 과제 설명

3. 느낀점

2. 과제 개요

- BMP 파일 헤더만 이용해서 BMP 파일을 읽고 픽셀의 값을 출력하는 프로그램을 작성

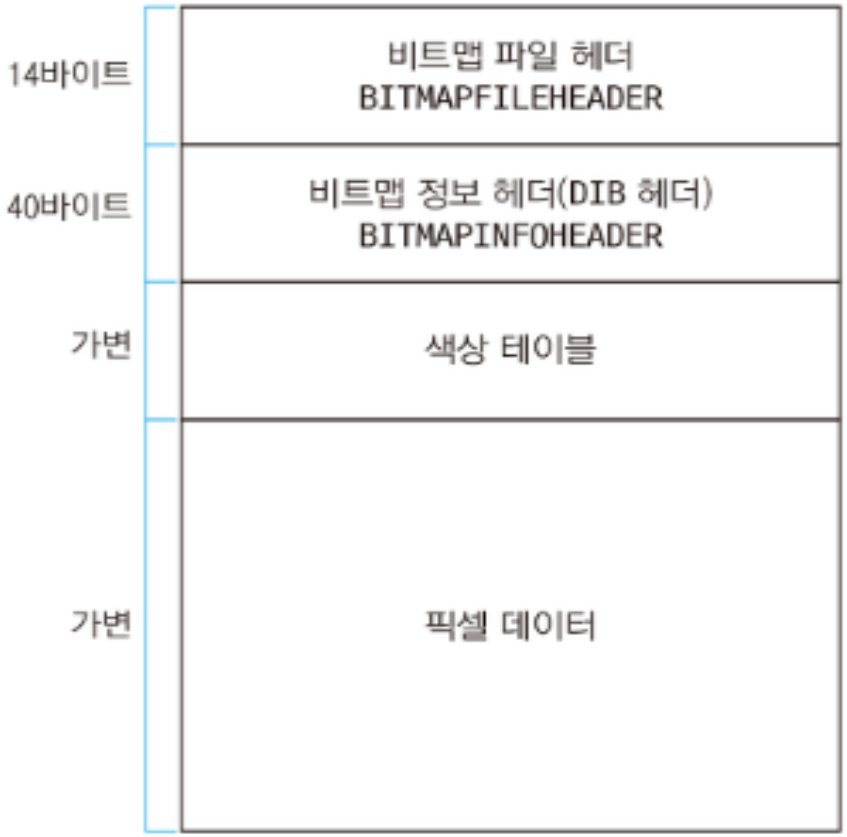
- BMP 파일 헤더만 이용해서 BMP 파일을 복사. 단, 복사되는 파일은 임의의 팔렛트 256개를 추가해야 합니다.

- 2중 for 루프를 1중 for 루프로 변경.

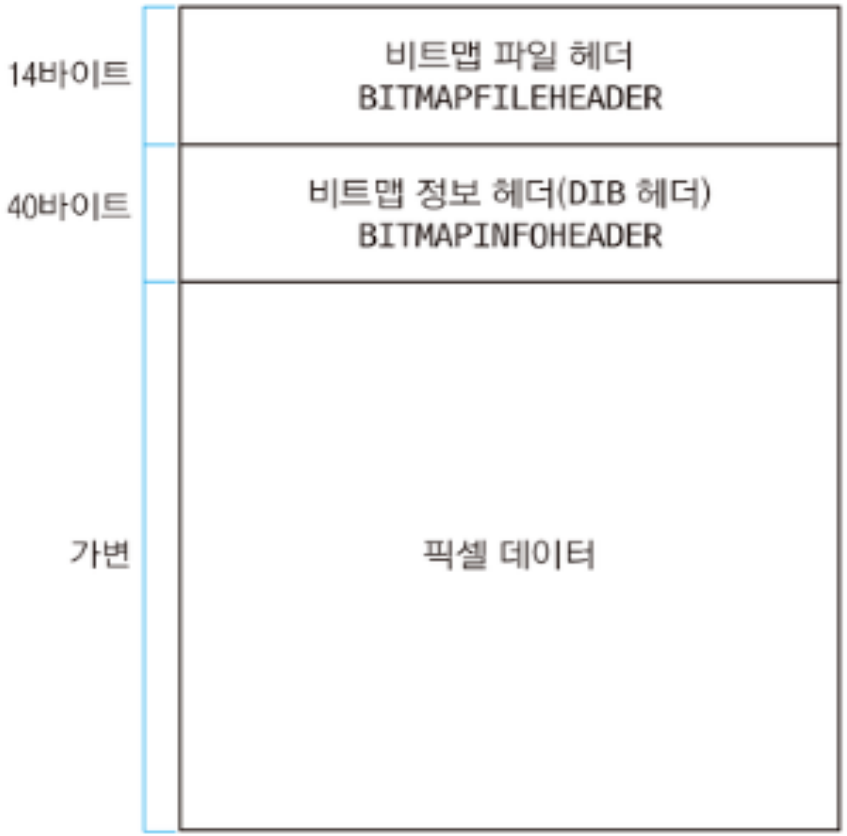
- 배열로 작성된 코드를 포인터로 변경.

3. 과제 설명

- BMP 파일 구조



픽셀당 1, 2, 4, 8비트



픽셀당 16, 24, 32비트

3. 과제 설명

- 픽셀의 값 출력

```
for(j = 0; j < bmpInfoHeader.biHeight ; j++) {
        for(i = 0; i < bmpInfoHeader.biWidth * elemsize ; i += elemsize) {
                B = inimg[i +j* bmpInfoHeader.biWidth * elemsize+0];
                G = inimg[i +j* bmpInfoHeader.biWidth * elemsize+1];
                R = inimg[i +j* bmpInfoHeader.biWidth * elemsize+2];

                printf("[B,G,R] : %d %d %d\n",inimg[i + j * width+0],
                                        inimg[i + j * width+1],
                                        inimg[i + j * width+2]);
        }
}
```

```
[B,G,R]  :  97 96 195
[B,G,R]  :  100 95 196
[B,G,R]  :  108 101 196
[B,G,R]  :  91 95 192
[B,G,R]  :  101 97 194
[B,G,R]  :  93 93 199
[B,G,R]  :  89 89 196
[B,G,R]  :  100 101 196
[B,G,R]  :  96 95 193
[B,G,R]  :  86 97 197
[B,G,R]  :  105 97 194
[B,G,R]  :  90 88 192
[B,G,R]  :  96 94 190
[B,G,R]  :  93 82 185
[B,G,R]  :  91 91 191
[B,G,R]  :  88 84 188
[B,G,R]  :  96 88 189
[B,G,R]  :  89 89 187
[B,G,R]  :  90 84 186
[B,G,R]  :  102 91 185
[B,G,R]  :  94 93 196
[B,G,R]  :  104 107 211
[B,G,R]  :  131 139 224
[B,G,R]  :  129 144 229
[B,G,R]  :  126 147 231
[B,G,R]  :  123 149 234
[B,G,R]  :  122 148 230
[B,G,R]  :  110 130 221
[B,G,R]  :  90 99 200
```

3. 과제 코드 설명

- 임의의 팔렛트 추가, 2중 for 루프 -> 1중 for 루프, 배열을 포인터로 변경

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "bmpHeader.h"

typedef unsigned char UBYTE;          //unsigned char형 UBYTE로 선언

int main(int argc, char** argv)

{
    FILE* fp;                                       //파일 포인터 선언
    UBYTE* inimg, * outimg;                         //입출력 파일을 unsigned char형으로 선언
    BITMAPFILEHEADER bmpFileHeader;                 //bmpHeader의 FILEHEADER 선언
    BITMAPINFOHEADER bmpInfoHeader;                 //bmpHedaer의 INFOHEADER 선언
    RGBQUAD* palrgb;

    if (argc != 3)                                  //명령인자의 개수가 3개가 아닐 경우 오류 출력하고 프로그램 종료
    {
        fprintf(stderr, "Usgae : Fail to file open\n");
        return -1;
    }


    if ((fp = fopen(argv[1], "rb")) == NULL)        //argv[1] 자리에 입력된 input 파일을 "rb : 바이너리파일 읽기전용"으로 열기
    {
        fprintf(stderr, "Usage : Fail to file open\n");
        return -1;
    }

    fread(&bmpFileHeader, sizeof(bmpFileHeader), 1, fp);                    //FILEHEADER 정보를 읽음
    fread(&bmpInfoHeader, sizeof(bmpInfoHeader), 1, fp);                    //INFOHEADER 정보를 읽음
```

3. 과제 코드 설명

```
int elemsize = bmpInfoHeader.biBitCount / 8;                          //input 파일(image)이 24비트인 경우 한 픽셀당 r,g,b 3가지 요소가 포함됨을 표현
int size = bmpInfoHeader.biWidth * elemsize;                          //image의 가로 사이즈 (r,g,b 개수 포함)
int imagesize = bmpInfoHeader.biWidth * bmpInfoHeader.biHeight * elemsize;    //image의 가로 * 세로 사이즈 (r,g,b 개수 포함)

inimg = (UBYTE*)malloc(sizeof(UBYTE) * imagesize);                    //input image를 사용하기 위한 메모리 할당
outimg = (UBYTE*)malloc(sizeof(UBYTE) * imagesize);                   //output image를 출력하기 위한 메모리 할당
palrgb = (RGBQUAD*)malloc(sizeof(RGBQUAD) * 256);                     //팔렛트 사용을 위한 메모리 할당

fread(palrgb, sizeof(RGBQUAD) * 256, 1, fp);                          //팔렛트 256개의 정보를 읽음
fread(inimg, sizeof(UBYTE), imagesize, fp);                           //input image를 imageSize 만큼 읽음

fclose(fp);                                                           //사용한 파일 포인터를 닫는다

for (int i = 0; i < 256; i++) {                                       //비어있는 팔렛트 채우기
    palrgb[i].rgbBlue = i;
    palrgb[i].rgbGreen = i;
    palrgb[i].rgbRed = i;
}

/* imageSize만큼 돌면서 inimg의 픽셀당 b,g,r 값을 outimg의 픽셀당 b,g,r 값에 넣어줌 */
for (int i = 0; i < imagesize; i += elemsize)
{
    //outimg[i+0] = inimg[i+0];
    //outimg[i+1] = inimg[i+1];
    //outimg[i+2] = inimg[i+2];
    *(outimg + (i + 0)) = *(inimg + (i + 0));
    *(outimg + (i + 1)) = *(inimg + (i + 1));
    *(outimg + (i + 2)) = *(inimg + (i + 2));
}
```

3. 과제 코드 설명

```
if ((fp = fopen(argv[2], "wb")) == NULL)                                    //argv[2] 자리에 입력된 output 파일을 "wb : 바이너리파일 쓰기전용"으로 열기
{
    fprintf(stderr, "File open fail\n");
    return -1;
}

fread(outimg, sizeof(UBYTE), imagesize, fp);                                //output image를 imageSize 만큼 읽음

fwrite(&bmpFileHeader, sizeof(bmpFileHeader), 1, fp);                       //FILEHEADER 정보를 읽음
fwrite(&bmpInfoHeader, sizeof(bmpInfoHeader), 1, fp);                       //INFOHEADER 정보를 읽음
fwrite(palrgb, sizeof(RGBQUAD) * 256, 1, fp);


bmpFileHeader.bfSize = sizeof(bmpFileHeader) + sizeof(bmpInfoHeader) + sizeof(RGBQUAD) * 256 + imagesize;   // 전체 사이즈를 FILEHEADER, INFOHEADER, 팔렛트의 크기만큼 지
정
bmpFileHeader.bfOffBits = sizeof(bmpFileHeader) + sizeof(bmpInfoHeader) + sizeof(RGBQUAD) * 256;           // 이미지 데이터를 넣기위해 Offset 이동
fwrite(outimg, sizeof(UBYTE), imagesize, fp);                               // output image를 imagesize 만큼 입력
free(inimg);                                                                // 할당된 메모리 해제
free(outimg);                                                               // 할당된 메모리 해제
free(palrgb);                                                               // 할당된 메모리 해제
fclose(fp);                                                                 // 사용한 파일 포인터 닫기

return 0;
}
```

4. 느낀점

- BMP 파일 구조에 대해 이해할 수 있었다.
- 팔렛트의 활용 방법에 대해 알 수 있었다.
- 오프셋을 통하여 파일 포인터에 대해 알 수 있었다.
- 배열과 포인터의 관계에 대해 알 수 있었다.