CS2310 Computer Programming, (15-16, Sem. A) Assignment 2

Due: Week 10, 4/12/2015, 5:00pm (No late submission will be accepted)

Important Notes:
Plagiarism is strictly prohibited in this course. Cases detected by the PASS system (programs with highly similarity) must be reported to the CS department and subjects to punishment. Last year, 29 cases were found and submitted.

1. Background

With the great success of assignment 1, you are going to enhance your program on ASCII art to the next level. In this assignment, you are going to convert a bitmap image into an ASCII art.

2. Bitmap

2.1 Bitmap File Header

Figure 1 and table 1 shows the file structure of bitmap. The first 14 bytes belong to the bitmap file header (table 2) that stores the general information about the bitmap. The first 2 bytes stores the signature of bitmap file, e.g. "BM", followed by the size of the image file. Position 10 (11th byte) is an integer that stores the starting location of pixel array. The pixel array stores the color values of each pixel in the image, row by row. The number of row and column can be found in the DIB header.

Figure 1. Bitmap file format:

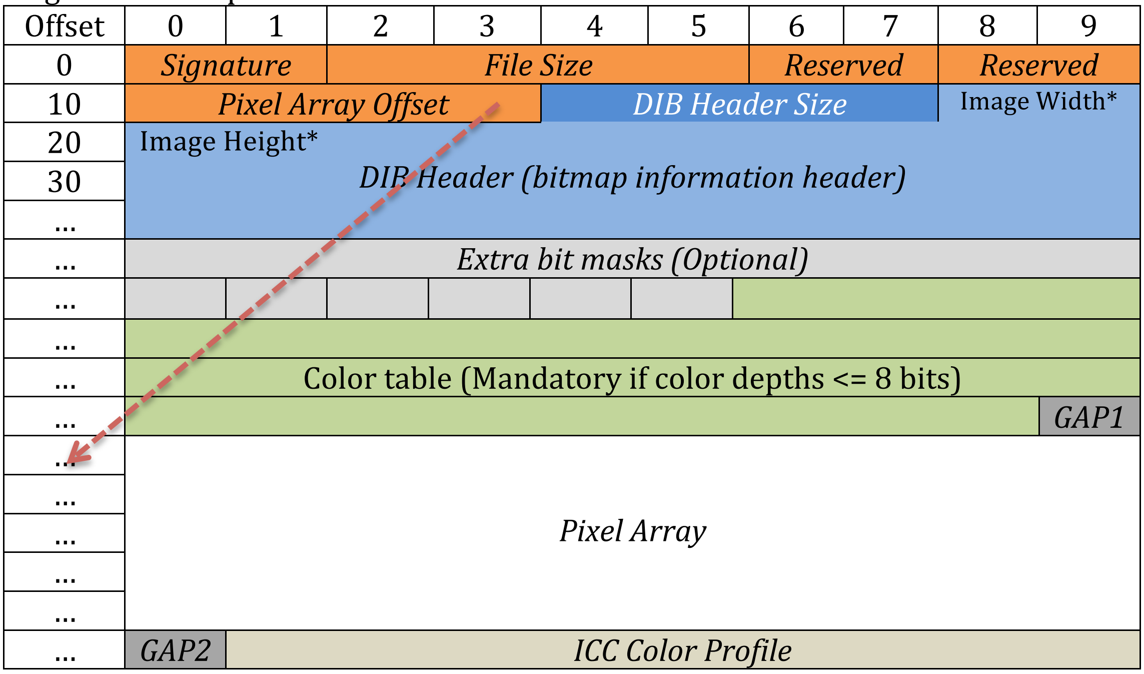


Table 1. Bitmap file structure

Name	Optional	Size (bytes)	Description
Bitmap file header	No.	14	See table 2 for detail
DIB header	No		See table 3a and 3b
Extra bit masks	Yes	12 or 16	
Color table	Yes	Variable	
Gap 1	Yes	Variable	Structure alignment
Pixel array	No	Variable	
Gap 2	Yes	Variable	Structure alignment
ICC Color profile	Yes	Variable	

Table 2. Bitmap file header

Offset	Size (bytes)	Description
0	2	Signature: Identify the BMP and DIB file "BM" – Windows Bitmap "BA"/"CI"/"CP"/"IC"/"PT" –OS/2 Bitmap and others
2	4	Bitmap file size in bytes
6	2	Reserved
8	2	Reserved
10	4	The starting address of bitmap image data

2.2 Bitmap Information Header (DIB Header)

The structure of DIB header is depended on the type of bitmap file, currently 7 possible types of bitmap file can be found. The DIB header size can be used to determine the type of bitmap file (table 3). The most common types are BITMAPCOREHEADER (table 3a) and BITMAPINFOHEADER (table 3b). The former one use short (16 bits) integer to store the image width and height while the latter one use int (32 bits).

Table 3. DIB Header Size

DIB Header Size	Header name	OS support
12	BITMAPCOREHEADER OS21XBITMAPHEADER	Window 2.0 or later OS/2 1.x
64	OS22XBITMAPHEADER	OS/2 BITMAPCOREHEADER2
40	BITMAPINFOHEADER	Windows NT, 3.1 or later
52	BITMAPV2INFOHEADER	Undocumented
56	BITMAPV3INFOHEADER	Undocumented
108	BITMAPV4INFOHEADER	Windows NT 4.0, 95 or later
124	BITMAPV5INFOHEADER	Windows NT 5.0, 98 or later

Table 3a. DIB Header (BITMAPCOREHEADER OS/2 1.x)

Offset	Size (bytes)	Description
14	4	DIB Header Size
18	2	Width in pixel (unsigned 16 bit)
20	2	Height in pixel (unsigned 16 bit)
22	2	Number of color planes, must be 1
24	2	The number of bits per pixel

Table 3b. DIB Header (BITMAPINFOHEADER)

Offset	Size (bytes)	Description
14	4	DIB Header Size
18	4	Width in pixel (signed int)
22	4	Height in pixel (signed int)
26	2	Number of color planes, must be 1
28	2	The number of bits per pixel
30	4	Compression method
34	4	Image size
38	4	Horizontal resolution of the image
42	4	Vertical resolution of the image
46	4	Number of colors in the color palette
50	4	Number of important colors used, usually ignored

2.3 Pixel Array

Pixel array is a place to store color value of each pixel. The size of each pixel is indicated in the DIB header, the field "the number of bits per pixel". E.g. 24 means 24 bits is used to store the color value, normally 8 bits for red, 8 bits for green and 8 bits for blue color components. Padding will be added to each row to make the total number of bytes is a multiple of 4. Data is stored in upside-down order, i.e. the first row of pixel array store the last row of image data if image height is positive. Negative height means the rows is stored from top to bottom.

For 24 bits pixel value, the first 8 bits store the red color component, and the second 8 bits store the green color component and the last 8 bits store blue.

3. Bitmap to ASCII art

You task is to write a program that accepts a string, which is the input bitmap file name, covert the color values of the image into an ASCII character and output to the screen. The conversation of ASCII character from color value is listed below:

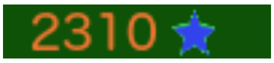
GrayLevel=0.3*R/256+0.6*G/256+0.11*B/256
If GrayLevel greater than 0.6, use uppercase ('A' to 'Z') to represent the gray level evenly. 'A' represents the lightest level and 'Z' represents the darkest level

If GrayLevel is between 0.3 and 0.6 (inclusively), use lower case ('a' to 'z') to represent the gray level.

If GrayLevel is less than 0.3, use (' ') to ('/') to represent the gray level.

You may assume the pixel format of Input bitmap is always in 24 bits pixels, 8R8G8B.

E.g.
Case 1:

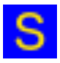


Sample Input:
2310.bmp

Sample Output:

```
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!m!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!"hih"!!!!!!""iih"!!!!!!"iii"!!!!!!!!"iih"!!!!!!!!!!!!!!!!P!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!"iih"hii!!!!"ii""hi"!!!!!"hii"!!!!!!"iiihii"!!!!!!!!!!!!!!lQm!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!hh!!!!"i"!!"i"!!!!"i"!!!!!!"i"!!!!!!"i"!!!"i"!!!!!!!!!!!!!PQQ!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!"i"!!!!!"h!!!"!!!!!"i"!!!!!!"i"!!!!!"i"!!!!!h"!!!!!!!!!!!!"QQQ"!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!"!!!!!!h"!!!!!!!!!"i"!!!!!!"i"!!!!!"i!!!!!!"i!!!!!!!!"kllnQQQnll"!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!"i"!!!!!!!!"hh!!!!!!!"i"!!!!!"h!!!!!!"i"!!!!!lmnPPQQQQQQPPnml!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!hi"!!!!!!"iii"!!!!!!!!"i"!!!!!""!!!!!!"i"!!!!!!oQQQQQQQQQQQQQP!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!"ii"!!!!!!!!""hi"!!!!!!!"i"!!!!!"h!!!!!!"i"!!!!!!!oQQQQQQQQQQQP!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!"i"!!!!!!!!!!!!!"i"!!!!!!"i"!!!!!"h!!!!!!"i"!!!!!!!!nPQQQQQQQPP!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!hi"!!!!!!!!!!!!!!!"h!!!!!!"i"!!!!!"i!!!!!!"i!!!!!!!!!!mQQQQQQQn!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!"i"!!!!!!!!""!!!!!!"h!!!!!!"i"!!!!!"i"!!!!!h"!!!!!!!!!!nQQQQQQQm!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!"i"!!!!!!!!!"i"!!!!"i"!!!!!!"i"!!!!!!"i"!!!"i"!!!!!!!!!!PQQQPQQQQ!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!"ih"""""""!!!"ih""iih!!!!!!!"i"!!!!!!"iihhii"!!!!!!!!!!!PQQPmPQQQ!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!"iiiiiiiii!!!!""iih"!!!!!!!!!""!!!!!!!!"iih"!!!!!!!!!!!"QPP!!!oPQ!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!mP!!!!!!lPm!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!m!!!!!!!!!o!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
```

Case 2:



Sample Input:
letter_s.bmp

Sample Output:

```
oooooooooooooooooooo
oooooooooooooooooooo
oooooooooooooooooooo
ooooookkjklnPPoooooo
oooonhjklljjlQPooooo
oooojinQooojjnQooooo
oooo"jQPooonhlQooooo
oooohjQPoooooooooooo
ooookikooPoooooooooo
oooookijjklnPPoooooo
ooooooomlkjjkPPooooo
ooooooooooolikRPoooo
ooooloPoooooijPPoooo
ooomhkRoooooijPPoooo
ooooijnQooolhkRPoooo
oooonijkllkjjPPooooo
ooooookkjjlmPPoooooo
oooooooooooooooooooo
oooooooooooooooooooo
oooooooooooooooooooo
```

4. Submission

Source program must submit to the PASS system on or before deadline. No report nor source printout is needed.

5. Guideline

Open a binary file

```
ifstream fin.open("filename.bmp",ios::binary);
```

Read two bytes of data and store the value in an integer

```
int x;
fin.read( (char*)&x,2);
```

Read four bytes of data and store the value in an integer

```
int y;
fin.read( (char*)&y,4);
```

To extract the second byte from an Integer

```
int rbg;
int blue;

blue=(rgb>>8)&0xff;
```

6.Reference

- http://www.fileformat.info/format/bmp/egff.htm
- https://msdn.microsoft.com/en-us/library/windows/desktop/dd183391(v=vs.85).aspx
- https://en.wikipedia.org/wiki/BMP_file_format