M1101 – M2101 Architecture et Système

Encodage des images BMP

DOLPHIN Nicolas

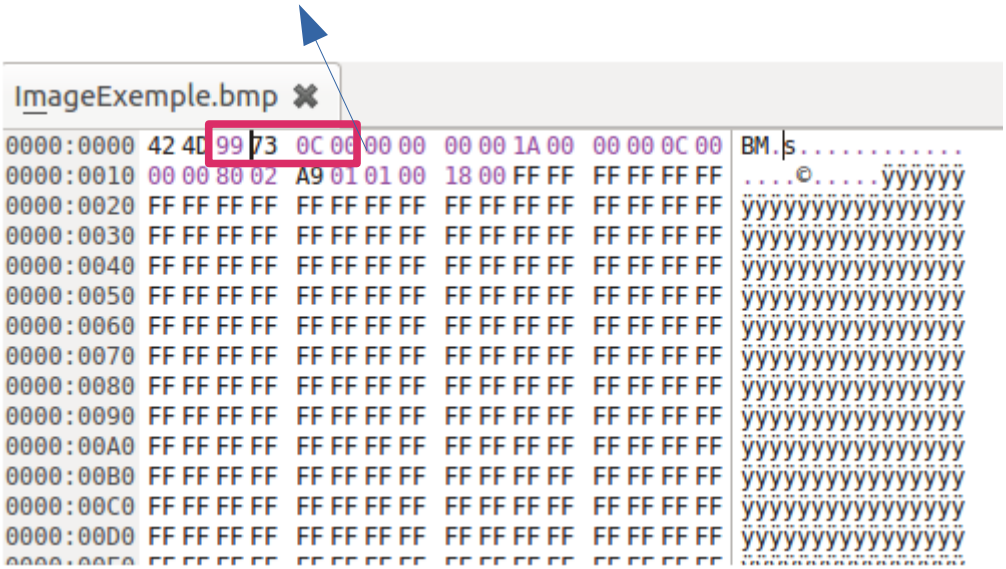
IUT d'Orleans

2018-2019

Question A.0

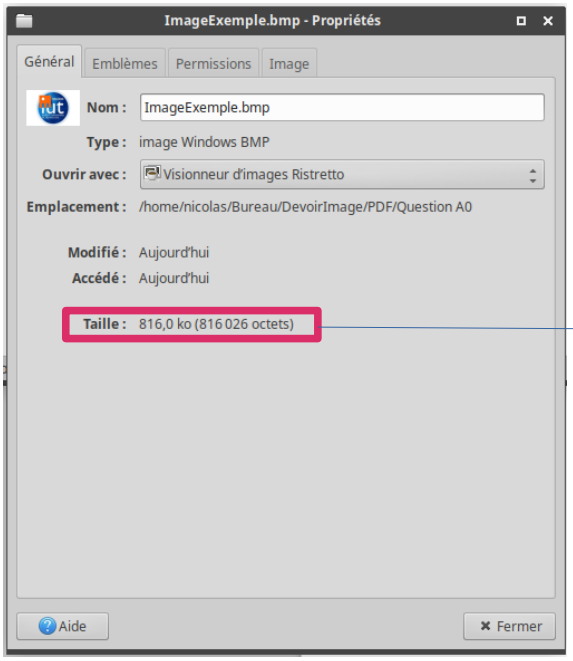
Erreur : display-im6.q16: length and filesize do not match `ImageExemple.bmp'...

Cela signifie donc que la taille du fichier ne correspond pas



000C7399 en Little Endian ce qui correspond donc à 816025 octets.

Or si on va dans les proprietes de l'image ImageExemple.bmp on a :



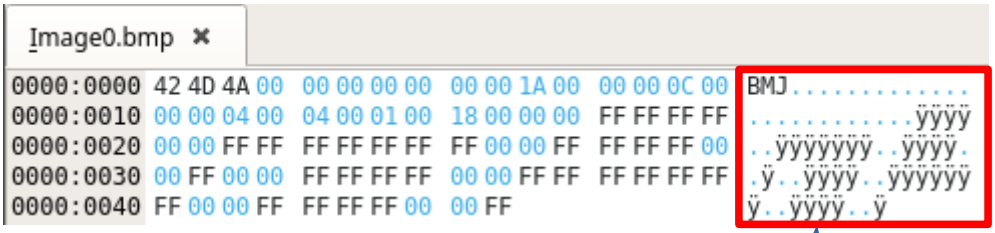
≠ 816025

Pour corriger l'erreur et faire en sorte qu'elle ne s'affiche plus il faut modifier la taille renseigner sur okteta par :

816026(base 10) en base 16 c'est a dire C739A et donc 9A730C en Little Endian

Il faudra par la suite renseigner la taille du fichier qu'une fois fini pour ne pas avoir l'erreur

Question A.1



L'entete (sur 14 octets) :

42 4D : En hexa corespond a BM comme on peut le voir sur la 3e colone de okteta

4A 00 00 00 :la taille du fichier BMP en octet (CF A.0)

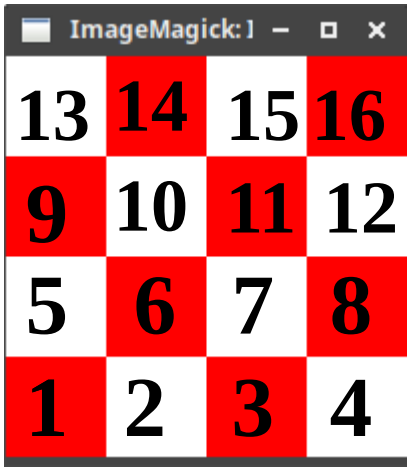
00 00 00 00 : octet libre → forcement des 0

1A 00 00 00 : décalage des bits de pixels = 26 octets pour atteindre le premier pixel

BITMAPCOREHEADER (sur 12 octets) :

0C 00 00 00 : taille de la structure = 12 octets
04 00 : largeur de l'image en pixels
04 00 : hauteur de l'image en pixels
01 00 : nombre de plans pour le rendu → toujours 1
18 00 : nombre de bits par pixels → 24 bits des couleur RGB soit 16 millions de couleurs

Les Pixels :



1,2,3,4,…,16 : Ordre de notations sur l'editeur hexadecimal

Rouge : #FF0000 (donc 0000FF en Little Endian)

Blanc: #FFFFFF (donc FFFFFF en Little Endian)

Question A.2

Imagetest.bmp

```
0000:0000 42 4D 4A 00  00 00 00 00  00 00 1A 00  00 00 0C 00  BMJ.............
0000:0010 00 00 04 00  04 00 01 00  18 00 FF FF  00 FF 00 FF  ..........ÿÿ.ÿ.ÿ
0000:0020 E8 9D 0F FF  FF FF 00 00  FF 00 00 FF  00 00 FF FF  è..ÿÿÿ..ÿ..ÿ..ÿÿ
0000:0030 00 00 00 00  FF 00 00 FF  00 00 FF 00  00 FF 00 00  ....ÿ..ÿ..ÿ..ÿ..
0000:0040 FF 00 FF 00  00 00 FF 00  00 FF                    ÿ.ÿ...ÿ..ÿ
```

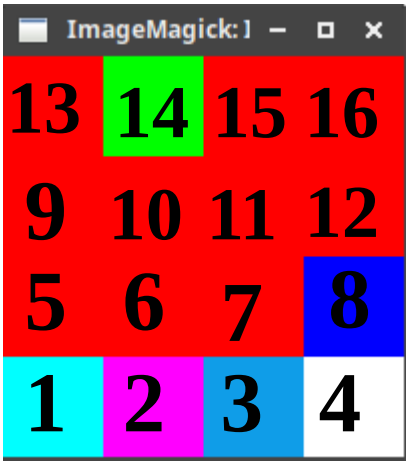
L'entete (sur 14 octets) :

Voir L'entete (sur 14 octets) de la question A.1 du damier rouge

BITMAPCOREHEADER (sur 12 octets):

Voir BITMAPCOREHEADER (sur 12 octets) de la question A.1 du damier rouge

Les Pixels :



1,2,3,4,…,16 : Ordre de notations sur l'editeur hexadecimal

Rouge : 0000FF en Little Endian

Vert: 0000FF en Little Endian

Blanc: FFFFFF en Little Endian

Bleu: FF0000 en Little Endian

Cyan: FFFF00 en Little Endian

Magenta: FF00FF en Little Endian

Bleu céruléen: E89D0F en Little Endian

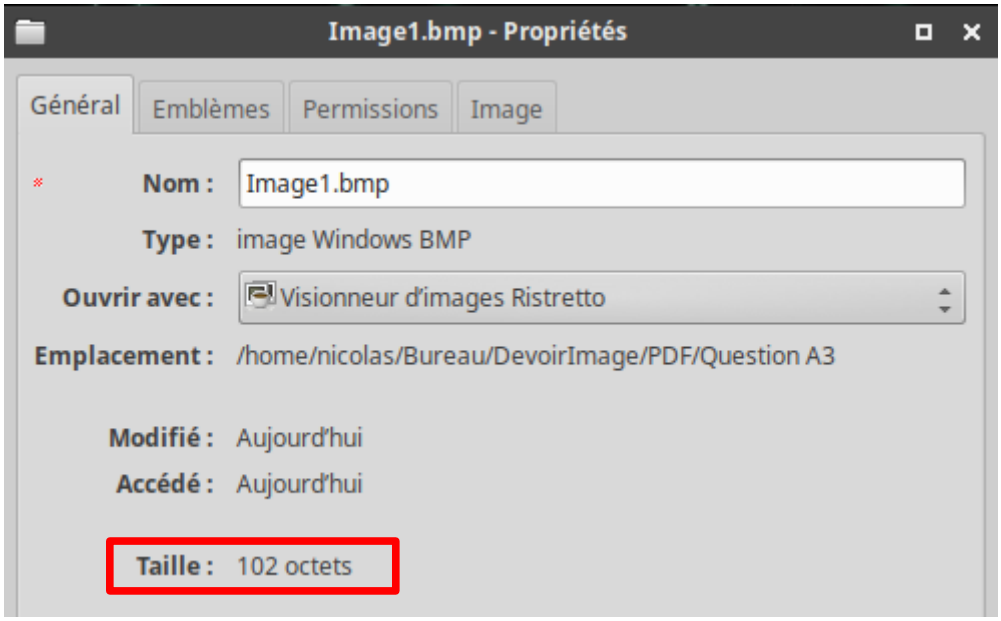
Question A.3

Le Poids

Le poids de la nouvelle image (Image1.bmp) doit etre de :

74 (taille d'origine)+ (40(BITMAPINFOHEADER)-12(BITMAPCOREHEADER))=102 octets

En enffet en allant dans les proprietes de Image1.bmp on a :



Diiferences

L'entete :

Taille du fichier (CF Le Poids) 4A000000 → 66000000

Décalage des bits de pixels 1A000000 → 36000000

BITMAPCOREHEADER (sur 12 octets) ***devenu*** *BITMAPINFOHEADER (sur 40 octets):*

taille de la structure (CF enonce) 0C000000 → 28000000

Largeur et Longueur de l'image en pixels 04000 → 04000000

Nouvelles Informations (Exemple type de compression, resolution, nombre de couleur , …)

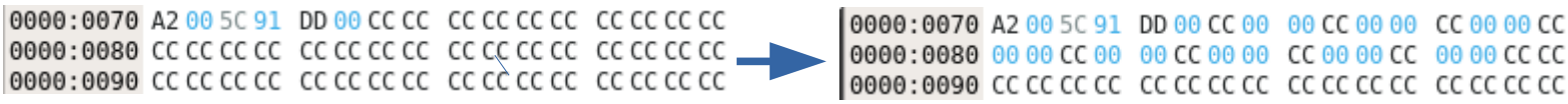
Autre Questions :

1. 18 00 → **24 bits/pixel** ce qui correspond au RVB avec ces 2^24 de couleurs
2. 48 octets (nombres d'octets codant tous les pixels)
3. Aucune Compression (BITMAPINFOHEADER donne l'info 00000000)
4. Aucune Difference pour le codage des pixels (toujours 3 octets et les memes sequence)

Question A.4

1. 01000000 → 1bits/pixel
2. 24 octets (nombres d'octets codant tous les pixels)
3. 00000000 (type de compression) → Aucune Compression
4. codées sur 4 octets avec la **Color table RVB** *en Little Endian (l'ordre bleu,vert, rouge)*
5. Il y a 2 couleurs dans la pallette le rouge(0000FF en Little Endian) et blanc(FFFFFF)
6. Oui, il sont maintenant codé sur 4 octets au lieu de 3 octets
7. Changer 0000FF par FF0000 a l'octet 55
8. Changer FF000000FFFFFF par FFFFFF00FF0000 a l'octet 55
9. CF Image3.bmp
10. CF ImageExempleIndexBMP3_16.bmp
11. On peut trouver le nombre de couleurs qu'il y a dans la palette a l'adresse 0000:002E
12. La couleur dominante "Blanc" utilisée par cette image se trouve a l'adresse 0000:0066
13. Le tableau de pixel commence a l'adresse 0000:0076
14. Le codage des pixel commence donc au 76e octets :

D'apres Oketea on voir que la sequence CCCCCC code la couleur blanc on sait donc que en Little Endian RGB CC0000 code la couleur bleu



15. Si l'on diminue le nombre de couleurs dans la palette :

    - Les couleurs se rapprochent du noir et du blanc (*au point de vue visuel*)
    - Seulement 4 couleurs sont code sur 10. En effet les 6 autres couleurs sont code par la valeur 0

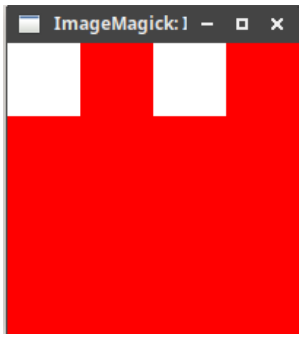
Quant a l'image elle-meme , on peut remarquer qu'elle se ternit grandement

CF ImageExempleIndexBMP3_4.bmp

Question A.5

Image3.bmp

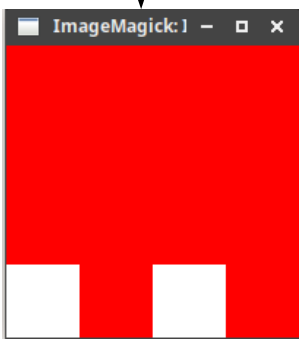
```
0000:0000 42 4D 4E 00  00 00 00 00  00 00 3E 00  00 00 28 00
0000:0010 00 00 04 00  00 00 03 00  00 00 01 00  01 00 00 00
0000:0020 00 00 10 00  00 00 00 00  00 00 00 00  00 00 02 00
0000:0030 00 00 02 00  00 00 00 00  FF 00 FF FF  FF 00 00 00
0000:0040 00 00 00 00  00 00 A0 00  00 00 50 00  00 00
```



(-4) revient a dire , FFFFFFFF – 4 ce qui nous donne donc FCFFFFFF en Little Endian

Image3.bmp

```
0000:0000 42 4D 4E 00  00 00 00 00  00 00 3E 00  00 00 28 00
0000:0010 00 00 04 00  00 00 FC FF  FF FF 01 00  01 00 00 00
0000:0020 00 00 10 00  00 00 00 00  00 00 00 00  00 00 02 00
0000:0030 00 00 02 00  00 00 00 00  FF 00 FF FF  FF 00 00 00
0000:0040 00 00 00 00  00 00 A0 00  00 00 50 00  00 00
```



On peut donc remarquer que si on change la hauteur du fichier par sa valeur negative il se passe une inversion de l'image

D'apres cette information on peut facilement inverser ImageExempleIndexBMP3_16.bmp

```
0000:0000 42 4D B6 13  02 00 00 00  00 00 76 00  00 00 28 00
0000:0010 00 00 80 02  00 00 A9 01  00 00 01 00  04 00 00 00
0000:0020 00 00 40 13  02 00 00 00  00 00 00 00  00 00 10 00
```

→

```
0000:0000 42 4D B6 13  02 00 00 00  00 00 76 00  00 00 28 00
0000:0010 00 00 80 02  00 00 57 FE  FF FF 01 00  04 00 00 00
0000:0020 00 00 40 13  02 00 00 00  00 00 00 00  00 00 10 00
```

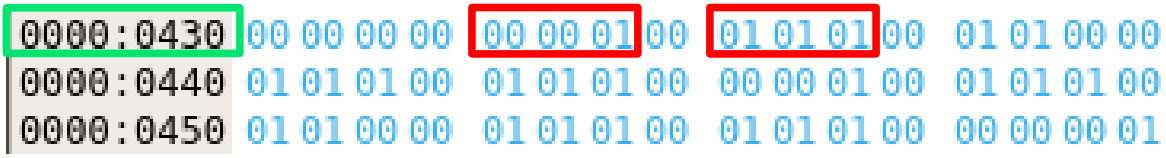
CF ImageExempleIndexBMP3_16.bmp

Question A.6

1. Le poids du fichier est de 1120 octets. Le poids a autant augmeter a cause de nombreux octets qui se sont rajouter lors de la compression. La compression a donc augmenté la taille du fichier au lieu de la diminuer

2. L'offset qui donne l'adresse de début des pixels est 0000:0430

3. Le coage des pixel se trouve a la fin de l'editeur , on a donc :



00 00 01 → correespond donc a la couleur Rouge

(B V R)

01 01 01 → correespond donc a la couleur Blanc

(B V R)

Et donc comme pour le codage de Image0.bmp on code les pixel en commencant en bas a gauche et en allant de gauche a droite.

Question A.7

1. le poids du fchier Image5.bmp est de 1102 octets. On peut expliquer sa taille plus petite que Image4.bmp par un codage de pixel plus petit

2.

04 00 code 4 pixels sont rouges

00 00 code la fin de ligne

Question A.8 , Question A.9 , Question A.10

Voir Methodologie de Question A.1 ou Question A.2