プログラミング発展

2023年度2Q　火曜日5~7時限(13:45~16:30)
金曜日5~7時限(13:45~16:30)

工学院　情報通信系

尾形わかは，松本隆太郎，
Chu Van Thiem, Saetia Supat
TA:東海林郷志，千脇彰悟

第11回「ファイル入出力2」

1. 前回の課題の解説
2. ファイルフォーマット
   ビットマップ（.bmp）を例に

ファイルフォーマット

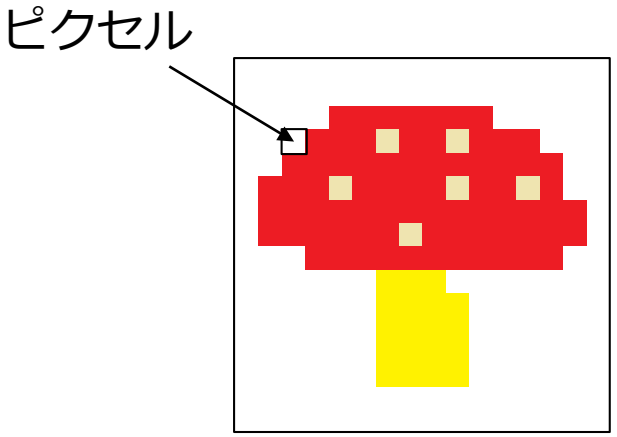
ビットマップを例に

ファイルフォーマット

- どのようなデータがどのように並んでいるのかを知らないと，適切にファイルを利用することができない.
- 一般的に利用されるファイルは，何がどのように並んでいるのかを，「ファイルフォーマット」として規定.
- 多種多様のファイルフォーマットのうち，どれを使っているのかを識別するのがファイルの「拡張子」.
  - 例）　.txt　⇒　文字が並んでいる
    .jpg　⇒　圧縮された画像 JPEG
- 各アプリケーションは，扱えるファイルフォーマットが決まっている.
  （テキストエディット → テキストファイル　とか）

ファイルフォーマットの例：ビットマップ .bmp

- 画像データのためのファイルフォーマットの一つ，効率は悪いが，単純で分かりやすい.
- 縦横に並ぶ「ピクセル」の色情報で画像を表現.
  - cf. ベクタ(ベクトル)データ



- ファイルの内容は主に、
  - ファイルヘッダ
  - 情報ヘッダ
  - ビットマップデータ

  からなる。

ファイルヘッダ（１４バイト）

拡張子が不明でも、この部分を見れば、
ファイルフォーマットを推定できる

オフセット	サイズ	格納する情報	値・備考
0x0000	2 バイト	ファイルタイプ	常に**BM (0x42, 0x4d)**
0x0002	4 バイト	ファイルサイズ	ビットマップファイルのサイズを格納する（単位はバイト）。
0x0006	2 バイト	予約領域1	常に0
0x0008	2 バイト	予約領域2	常に0
0x000a	4 バイト	オフセット	ファイルヘッダの先頭アドレスからビットマップデータの先頭アドレスまでのオフセット（単位はバイト）。

画像のデータ本体を読むには、このサイズ分を読み飛ばせばよい

それぞれの値を知りたければ、ファイルの頭（SEEK__SET）から
オフセット分だけ後ろに移動して（fseek）から読み込めばよい

https://ja.wikipedia.org/wiki/Windows_bitmap

情報ヘッダ（BITMAPINFOHEADER）

（40バイト）

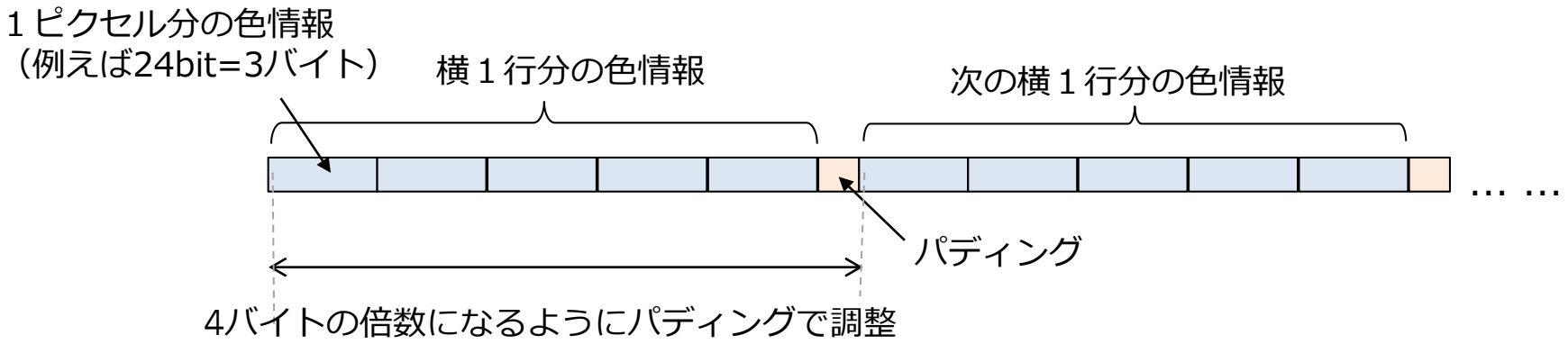
オフセット	サイズ	格納する情報	値・備考
0x000e	4 バイト	ヘッダサイズ	40
0x0012	4 バイト	ビットマップの横幅	単位はピクセル
0x0016	4 バイト	ビットマップの縦幅	単位はピクセル。値が負の場合はトップダウン画像となる
0x001a	2 バイト	プレーン数	常に1
0x001c	2 バイト	1ピクセルあたりのビット数	0,1,4,8,16,24,32
0x001e	4 バイト	圧縮形式	0,1,2,3,4,5 ※1
0x0022	4 バイト	画像データサイズ	単位はバイト
0x0026	4 バイト	水平方向の解像度	単位はピクセル/m
0x002a	4 バイト	垂直方向の解像度	単位はピクセル/m
0x002e	4 バイト	使用する色数	ビットマップで実際に使用するカラーパレット内のカラーインデックスの数。
0x0032	4 バイト	重要な色数	ビットマップを表示するために必要なカラーインデックスの数。

https://ja.wikipedia.org/wiki/Windows_bitmap

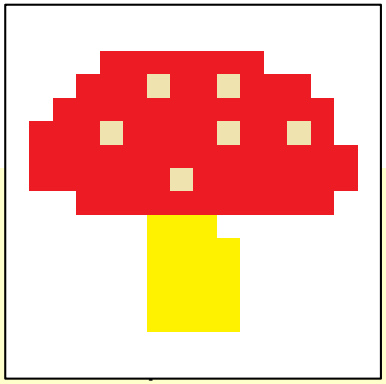
ビットマップデータ

- 左下のピクセルから右へ，順に各ピクセルの色情報を並べてある．（※トップダウンでない場合．）
- 各ピクセルの色情報を表現するのに使うビット長は，ヘッダに記載されている．（0, 1, 4, 8, 16, 24, 32）
  - 24：BGRの順で各色8ビット＝１バイト（256階調）．フルカラー

- ただし、横１行分のデータのバイト数が４の倍数で無い場合は，0でパディングすることで，必ず４の倍数とする．



ビットマップの例（16×16ピクセル）

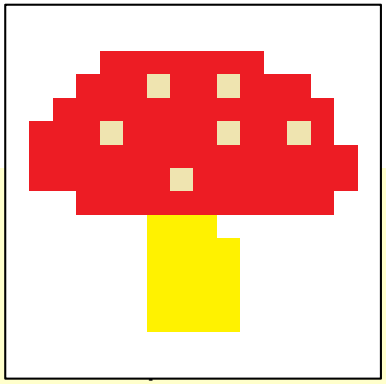


BITMAPFILEHEADERの部分

```
42 4d 36 03 00 00 00 00 00 00 36 00 00 00 28 00   | BM6         6   (
00 00 10 00 00 00 10 00 00 00 01 00 18 00 00 00   |
00 00 00 03 00 00 00 00 00 00 00 00 00 00 00 00   |
00 00 00 00 00 00 ff ff ff ff ff ff ff ff ff ff   |
ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff   |
ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff   |
ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff   |
ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff   |
```

オフセット	サイズ	格納する情報	値・備考
0x0000	2 バイト	ファイルタイプ	常に**BM (0x42, 0x4d)**
0x0002	4 バイト	ファイルサイズ	ビットマップファイルのサイズを格納する（単位はバイト）。
0x0006	2 バイト	予約領域1	常に0
0x0008	2 バイト	予約領域2	常に0
0x000a	4 バイト	オフセット	ファイルヘッダの先頭アドレスからビットマップデータの先頭アドレスまでのオフセット（単位はバイト）。

ビットマップの例（16×16ピクセル）



BITMAPFILEHEADERの部分

```
42 4d 36 03 00 00 00 00 00 00 36 00 00 00 28 00   | BM6       6   (
00 00 10 00 00 00 10 00 00 00 01 00 18 00 00 00   |
00 00 00 03 00 00 00 00 00 00 00 00 00 00 00 00   |
00 00 00 00 00 00 ff ff ff ff ff ff ff ff ff ff   |
ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff   |
ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff   |
ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff   |
ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff   |
```

オフセット	サイズ	格納する情報	値・備考
0x0000	2バイト	ファイルタイプ	常に**BM (0x42, 0x4d)**
0x0002	4バイト	ファイルサイズ	ビットマップファイルのサイズを格納する（単位はバイト）。
0x0006	2バイト	予約領域1	常に0
			アドレス
			タの先頭
			ット（単

ヘッダ内のすべての値は，little endianで記載されている.

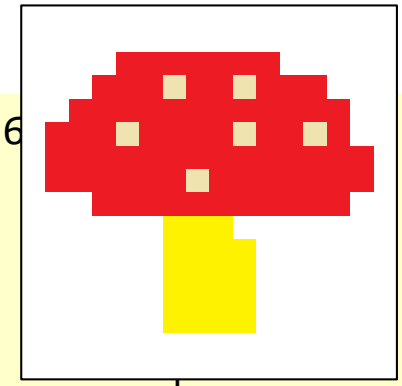
- ファイルサイズ　：　36 03 00 00 → 0x00000336
- オフセット　：　36 00 00 00 → 0x00000036

リトルエンディアンCPUを仮定すると、
freadで適切なサイズの整数型変数に読み込めばよい.

ビットマップの例（16×16ピクセル）

BITMAPINFOHEADER

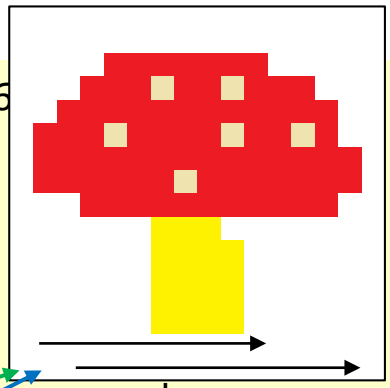
```
42 4d 36 03 00 00 00 00 00 00 36 00 00 00 28 00 | BM6         6   (
00 00 10 00 00 00 10 00 00 00 01 00 18 00 00 00 |
00 00 00 03 00 00 00 00 00 00 00 00 00 00 00 00 |
00 00 00 00 00 00 ff ff ff ff ff ff ff ff ff ff |
ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff |
ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff |
ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff |
ff
ff
```



オフセット	サイズ	格納する情報	値・備考
0x000e	4 バイト	ヘッダサイズ	40
0x0012	4 バイト	ビットマップの横幅	単位はピクセル
0x0016	4 バイト	ビットマップの縦幅	単位はピクセル。値が負の場合はトップダウン画像となる
0x001a	2 バイト	プレーン数	常に1
0x001c	2 バイト	1ピクセルあたりのビット数	0,1,4,8,16,24,32
0x001e	4 バイト	圧縮形式	0,1,2,3,4,5 ※1
0x0022	4 バイト	画像データサイズ	単位はバイト
0x0026	4 バイト	水平方向の解像度	単位はピクセル/m

ビットマップの例（16×16ピクセル）

```
42 4d 36 03 00 00 00 00 00 00 36 00 00 00 28 00 | BM6       6
00 00 10 00 00 00 10 00 00 00 01 00 18 00 00 00 |
00 00 00 03 00 00 00 00 00 00 00 00 00 00 00 00 |
00 00 00 00 00 00 ff ff ff ff ff ff ff ff ff ff |
ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff |
ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff |
ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff |
ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff |
ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff |
```



左下角のピクセルの色情報（24ビット＝3バイト）

左下角のすぐ右のピクセルの色情報

この画像では,
下のほうが白（ＲＧＢ全てが最大値のff）なので,
しばらくはひたすらffが並んでいます

補足

- BITMAPINFOHEADERの部分には、別のフォーマットのヘッダがくる場合もある。
  ヘッダによってヘッダサイズが異なるため、ヘッダサイズ＝40 の場合はBITMAPINFOHEADERと判別できる。

- 1ピクセルあたりのビット数が24,32以外のときは，情報ヘッダとビットマップデータの間に「カラーパレット」の情報が追加される.
  - 例） 1ピクセルあたりのビット数= 1 の場合，色情報は各ピクセルにつき 0 or 1 .
    0→白(r=255,g=255,b=255)，1→黒(r=0,g=0,b=0) といった対応関係を表すのがカラーパレット.

- ビットマップの縦幅が負の値の場合，トップダウンとなる．下から順に・・・ではなく，上から順に色情報が並ぶ.

fseekの利用例

- 画像ファイル fig.bmp から、画像のサイズ（縦と横のピクセル数）を知る：

```
fp = fopen("fig.bmp","rb");
fseek(fp, 0x0012, SEEK_SET);
fread(&x_size, 4, 1, fp);  // int x_size
fread(&y_size, 4, 1, fp);  // int y_size
```

この間で，いろいろ読み込んでいてもok

- サイズがわかった上で、画像データを読み込む：

```
fseek(fp, 0x0a, SEEK_SET);
fread(&data_offset, 4, 1, fp);   // int data_offset
            // data_offset = ビットマップデータの始まり位置.
fseek(fp, data_offset, SEEK_SET);
for  (y=0; y<y_size; y++)
    for (x=0; x<x_size; x++)   {
        fread(&color[x][y].b, 1, 1, fp);
        fread(&color[x][y].g, 1, 1, fp);
        fread(&color[x][y].r, 1, 1, fp);
    }
```

適当な構造体を仮定しています

行の最後に，パディングの分をfseekで読み飛ばす必要あり.

Ex11-1

（ビットマップから画像の読み込み）

①ファイル名を標準入力から受け取り、以下を行うプログラムを作りなさい。

1. ファイルがfopenできなければ，"Can not open."を出力して終了。
2. ファイルから必要情報を読み込み，以下の3つ全てが満たされていることを確認する.
   - ファイルの先頭の2バイトが"BM"であること。
   - 1ピクセルあたりのビット長が24であること。
   - **BITMAPFILEHEADER**内のオフセットが54であること。

   満たされていれば，"OK."と出力して終了．満たされていない場合は，満たされていない項目の数を出力した後、"Different file type."と出力して終了.

★提出物：ソースファイル
（ ex11_1_1.c ）

実行例：

```
filename ? : sample.bmp
OK.

filename ? : sample.txt
Number of unsatisfied items = 3.
Different file type.
```

Ex11-1　（つづき）

②ファイル名を標準入力から受けとり、以下を行うプログラムを作りなさい。

1. ファイルが開けなければ " Can not open."を出力して終了。
2. ①と同様にファイルのタイプをチェックし，タイプが異なる場合は "Different file type."と出力して終了。
3. ビットマップの横幅（ピクセル数）と 縦幅（ピクセル数）を読み込み、表示する。（縦幅が正の値であると仮定してよい．）
4. ビットマップの、左上の角、右上の角、左下の角、右下の角の、それぞれのピクセルの色情報を表示する。色情報は、ＲＧＢそれぞれの値を0～255の値で表示する。

※　step２のチェックは，①で作ったプログラムを関数にして，それを呼び出すとよい．（次の課題でも使える）

★提出物：ソースファイル　（ ex11_1_2.c ）

Ex11-1　（つづき）

実行例：

```
file name ? : sample.bmp

Horizontal size = 128, Vertical size = 128,
Upper-left (R=0, G=200, B=11)
Upper-right (R=32, G=113, B=256)
Lower-left (R=255, G=255, B=255)
Lower-right (R=0, G=0, B=0)
```

注意！

- 「横１行分のデータ長」のバイト数が４の倍数でない場合は，パディングがあるため，計算を間違えないこと.
- 色情報はＢ Ｇ Ｒの順で保存されている.

Ex11-2

（画像を加工する）
入力されたファイル名の画像ファイルから，画像を180度回転させた画像ファイルを作成するプログラムを作りなさい.

- 入力されたファイル名のファイルが開けなかったり，タイプが異なる場合は，Ex11-1と同様に"Different file type."と出力して終了すること.
- どんなサイズの画像でも（メモリが許す限り）動作するようにすること.
- 回転後の画像データは、自身の学籍番号を用いて"20B12345_ex11_2.bmp”というファイルに出力すること.

提出物：ソースファイル（ ex11_2.c ）

※　1ピクセルあたりのビット数が小さい場合，カラーパレットが必要になる．ただし，この課題では，色の情報は変更しないため，ビットマップデータまでのオフセット値をチェックし，カラーパレットを読み飛ばすことで，同様に処理は可能ではある.

Ex11-2　（補足説明）

画像を保存するための構造体をどうするか？
一例を挙げる.

1ピクセル分の色情報（ＲＢＧの値）は、次の構造体に保存.

```
typedef struct {
    unsigned char b;
    unsigned char g;
    unsigned char r;
} rgb_t;
```

3色まとめてunsigned char pix[3] でもよいかも.

画像全体の色情報は、可変長の二次元配列に保存.
縦横のサイズ x_size, y_size を読み込んでから

```
rgb_t fig[x_size][y_size];
```

Ex11-2　（発展課題）

③実行時の引数（argv, argc）を使える人は挑戦してみてください。
（argv, argcについては、12回目の講義で扱います。）

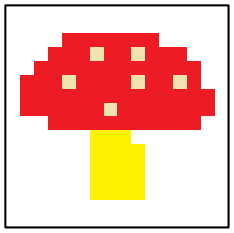
- ファイル名を実行時の引数としても与えられるプログラムにしてみよう．引数が無い場合は，ファイル名の入力を促すこと．
- さらに、入力画像の90度回転（右回転・左回転）もできるプログラムを作ってみよう．
  実行時の引数として、ファイル名と -right を与えると右90度回転した画像が、ファイル名と -left を与えると左90度回転した画像が、"19B12345_ex11_2.bmp”（自分の学籍番号にすること）に出力されるようにする．
  実行時に引数が無い場合は（argc==1），ファイル名の入力を促し、180度回転とすること．

（ ex11_2.c　として提出してよい．）

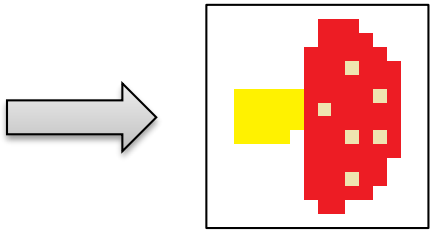
Ex11-2　（発展課題　続き）

実行例：

```
> ./a.out sample.bmp -right
Successfully rotated.
>
```



sample.bmp



19B12345_ex11_2.bmp

注意！
90度回転をしても、画像の全ピクセル数は変わらない.
しかし、縦横のサイズが逆転するため、パディングの有無が変わる可能性があり、画像データサイズが変わる可能性がある

今日の課題の提出物まとめ

- Ex11-1
  - ①　ex11_1_1.c
  - ②　ex11_1_2.c

- Ex11-2 :
  - ex11_2.c　　（発展課題の機能を付けたら加点）

画像ファイルの例：
　sample_A.bmp
　sample_B.bmp
をT2SCHOLAに置いたので，試してみること。
Bの方でも正しく動くように。

注意！

- fopenしたファイルポインタは（一回だけ）fcloseすること。
  fcloseで閉じていないファイルを再度fopenしないこと。

- freadでファイルから読み込む場合、読み込むバイト数と、読み込む先の変数のバイト数に気を付けよう．一致していれば問題ないが
  　int x; short y;
  のとき
  fread(&x, 2, 1, fp);　→　下位2バイトだけが上書きされ、上位２バイトは不変（初期化していなければ何が入っているかは不明）
  fread(&y,4,1,fp);　→　yに後ろのメモリにも書き込みしてしまう（不適切なメモリアクセス）

- 大きな画像を扱おうとすると、スタック領域にビットマップデータを保存するだけのメモリを確保できないかもしれません．課題では、サンプル画像程度のサイズの画像が扱えればOKとします。

コピペレポートについて

プログラムや考察などが他の提出者と重複している場合、不正とみなして減点および問い合わせをすることがあります