プログラミング演習 II
課題4第3週

iioLoadFile()と
iioMallocImageBuffer()の補足

iioLoadFile()

```
/*
        int iioLoadFile(IMAGE *pImage, const char *fname)

        画像ファイルをオープンし，
        適切なサイズの画像バッファを確保し
        IMAGE構造体に割り当てた後に，
        画像バッファにデータをロードする．

        Image *pImage       :   IMAGE構造体へのポインタ
        const char *fname   :   入力ファイル名
        戻り値              :   0(ファイルのロード成功)
                                1(ファイルのロード失敗)
*/
int iioLoadFile(IMAGE *pImage, const char *fname)
{
        FILE    *fpr;
        char    lineBuffer[LINEMAX];
        int     i;
```

```
/* ファイルオープン */                    "rb"は、read biinary
if ((fpr = fopen(fname, "rb")) == NULL) {
        perror(fname);
        return 1;
}

/* ヘッダ部読み込み開始 */
fgets(lineBuffer, LINEMAX, fpr);

if (strcmp(lineBuffer, "P6¥n")) {
        // ERROR
        fclose(fpr);
        return 1;
}

fgets(lineBuffer, LINEMAX, fpr);

/* コメント行の読み飛ばし */
while (lineBuffer[0] == '#')
        fgets(lineBuffer, LINEMAX, fpr);
```

	長さ	内容
ヘッダー	2バイト	P6
	1バイト	改行コード
	不定	横幅を示すアスキー文字列
	不定	セパレーター(空白)
	不定	縦幅を示すアスキー文字列
	1バイト	改行コード
	不定	最大輝度を表すアスキー文字
	1バイト	改行コード
データ	不定	輝度を示す値 (バイナリ)
	不定	輝度を示す値 (バイナリ)
	…	…

```
        /* IMAGE構造体のメンバ変数に，画像の幅，高さをセット．*/
        sscanf(lineBuffer, "%d %d¥n", &pImage->xsize, &pImage->ysize);

        fgets(lineBuffer, LINEMAX, fpr);

        /* IMAGE構造体のメンバ変数に，画像の最大輝度をセット．*/
        sscanf(lineBuffer, "%d", &pImage->level);

        /* ヘッダ部読み込み終了 */

        /* バッファ用メモリ確保 */
        iioMallocImageBuffer(pImage);

        /* バッファに画像データを格納 */
        for (i = 0; i < pImage->ysize; i++){        freadを画像の行数分だけ繰り返す
                fread(pImage->pBuffer[i], sizeof(PIXEL), pImage->xsize, fpr);
        }                                           fwriteも使い方は同じ

        /* ファイルクローズ */
        fclose(fpr);
        return 0;
}
```

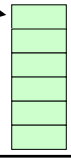
iioSaveFile()の作り方

```
int iioSaveFile(IMAGE *pImage, const char *fname)
{
    FILE *fpw; ←ファイル出力用ファイルポインタ
    int i;   ←画像の行数だけ繰り返すループ変数

    /* ファイルオープン*/ ←iioLoadFileと同じ。但し、"rb"⇒"wb"
                                        fpr⇒fpw
    /* ヘッダ出力 */
    fprintf(fpw,"P6¥n%d %d¥n%d¥n",
        pImage->xsize,pImage->ysize,pImage->level);

    /* 画像データをファイルへ出力 */  ←iioLoadFileと同じ。
                                      但し、fread⇒fwrite
                                            fpr⇒fpw
    fclose(fpw);
    return 0;
}
```

P6
xsize ysize
level

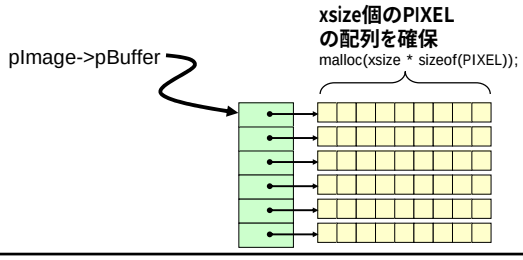
iioMallocImageBuffer()

```
void iioMallocImageBuffer(IMAGE *pImage)
{
        int i; ←画像の行数だけ繰り返すループ変数

        pImage->pBuffer = malloc(pImage->ysize*sizeof(PPIXEL));
```



ysize個のPPIXEL
の配列を確保
malloc(ysize * sizeof(PPIXEL));

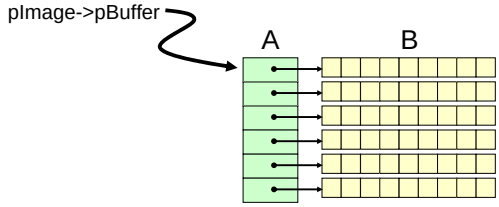
```
        for (i = 0; i < pImage->ysize; i++){
                pImage->pBuffer[i] = malloc(pImage->xsize * sizeof(PIXEL));
        }
}
```



iioFreeImageBuffer()の作り方

```
void iioFreeImageBuffer(IMAGE *pImage)
{
   int i; ←画像の行数だけ繰り返すループ変数
```

方針：Bを開放してからAを開放する



Bを開放

```
    for (i = 0; i < pImage->ysize; i++){
        pImage->pBuffer[i] = malloc(pImage->xsize * sizeof(PIXEL));
        free(pImage->pBuffer[i]);
    }
```

Aを開放

```
    free(pImage->pBuffer );
```