モジュール設計

現時点のプログラムの構造

- main.cに全ての変数と関数を定義している
  - 可読性低い，毎回全てコンパイル必要，秘匿性低い

```
#include ...
#define ...
構造体の定義
関数プロトタイプ宣言

int main(int argc, char *argv[])
{
   ...
}

iioLoadFile関数の定義
iioSaveFile関数の定義
iioMallocImageBuffer関数の定義
iioFreeImageBuffer関数の定義
ipCopy関数の定義
```

iioLoadFileにしか関係しない定数

ファイルの読み書きや
領域の確保解放に関係するもの

画像処理（画素値の編集）に関係するもの

モジュール設計

- 機能ごとに独立したファイルに分割する
  - 可読性の向上，分割コンパイル可能，情報の秘匿

main.c
画像ファイルをロードする関数を呼び出す
画像処理関数を呼び出す
画像ファイルをセーブする関数を呼び出す
バッファを解放する

img_io.c
画像ファイルのロード関数の定義
画像ファイルのセーブ関数の定義
画素値用バッファを確保する関数の定義
画素値用バッファを解放する関数の定義

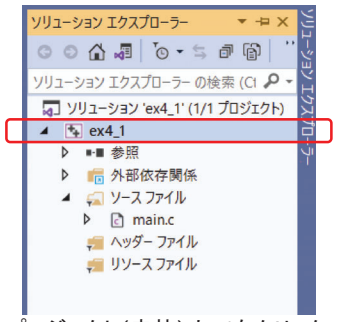
img_proc.c
画像処理のための関数の定義

モジュール設計の方針

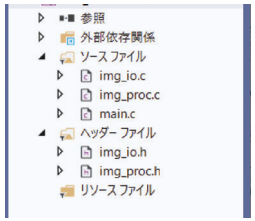
- main
  - 各ヘッダファイルのinclude
  - main関数
- img_io
  - PIXEL構造体およびIMAGE構造体の定義
  - iioLoadFile，iioSaveFile関数
  - iioMallocImageBuffer，iioFreeImageBuffer関数
- img_proc
  - ipCopy関数
  - その他の画像処理関数

【作業課題1/6】ファイルの追加

- main.cと同フォルダに以下のcファイルとヘッダファイルを新規に作成し，プロジェクトに追加
  - img_io.c
  - img_io.h
  - img_proc.c
  - img_proc.h



プロジェクト（赤枠）上で右クリック
→「追加」→「新しい項目」
でcファイルとhファイルを一つずつ追加する



追加後，こうなればOK

【作業課題2/6】img_io.hの作成

インクルードガードを追記する．末尾は自身の学籍番号にすること．
#pragma onceがある場合は削除する．

main.c

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

typedef struct {
    int    xsize;
    int    ysize;
    int    level;
    PIXEL **pBuffer;
} IMAGE;

int  iioLoadFile(IMAGE *pImage, const char *fname);
int  iioSaveFile(IMAGE *pImage, const char *fname);
void iioMallocImageBuffer(IMAGE *pImage);
void iioFreeImageBuffer(IMAGE *pImage);
```

main.cから構造体定義と関数プロトタイプ宣言(ファイル入出力とメモリ確保解放)を移動させる

#endif忘れずに

img_io.h

```
#ifndef IMG_IO_123456A
#define IMG_IO_123456A

#endif
```


【作業課題3/6】img_io.cの作成

_CRT…の定義と，stdlib.h，stdio.h，string.hのincludeはmain.cからコピペ

img_io.hを新たにinclude

LINEMAXの定義はmain.cから移動

main.c

```
int iioLoadFile(IMAGE *pImage, const char *fname)
{ ....
}

int iioSaveFile(IMAGE *pImage, const char *fname)
{ ....
}

void iioMallocImageBuffer(IMAGE *pImage)
{ ....
}

void iioFreeImageBuffer(IMAGE *pImage)
{ ....
}
```

ファイル入出力とメモリ確保解放の4つの関数定義をここに移動

img_io.c

```
#define _CRT_SECURE_NO_DEPRECATE
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include "img_io.h"

#define LINEMAX 100
```


【作業課題4/6】img_proc.hの作成

インクルードガードを追記する．末尾は自身の学籍番号にすること．
#pragma onceがある場合は削除する．

main.c

```
void ipCopy(IMAGE* p_in_image, IMAGE* p_out_image);
```

main.cからipCopy関数のプロトタイプ宣言を移動させる

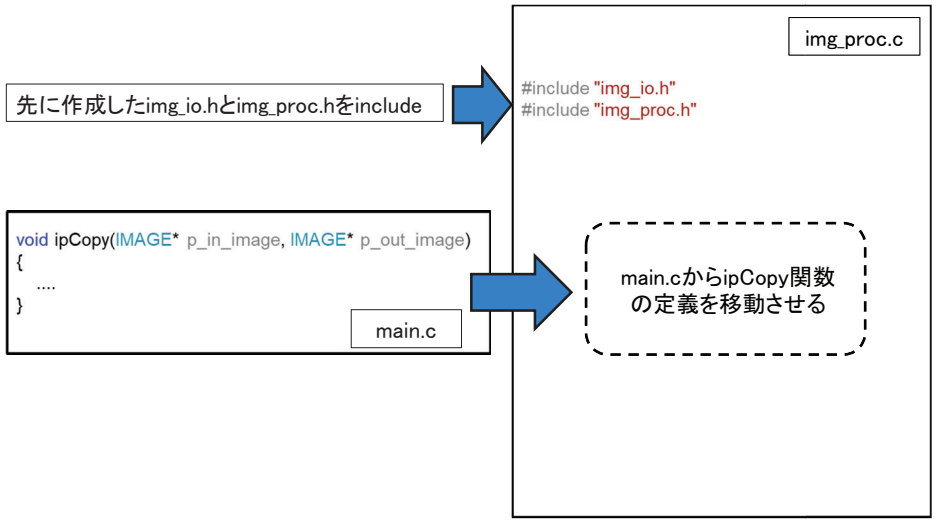
#endif忘れずに

img_proc.h

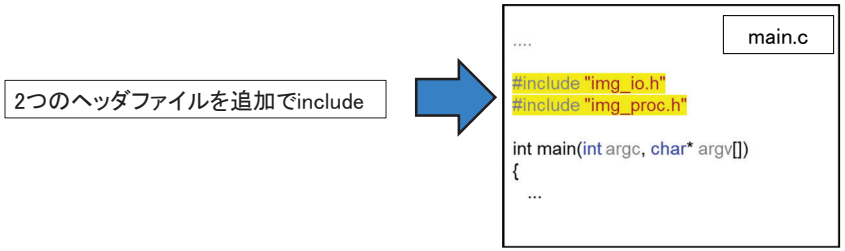
```
#ifndef IMG_PROC_123456A
#define IMG_PROC_123456A

#endif
```

【作業課題5/6】img_proc.cの作成



【作業課題6/6】main.cの修正



main.cではiioFreeImageBuffer関数やipCopy関数を呼び出す必要があるため，どちらのヘッダも必要になる．

モジュール化の確認

- これでモジュール化が完了したので，実行する
  - これまでと同じ結果が得られるか確認する
  - また，main.cがいかにコンパクトになったか確認する

main.c
画像ファイルをロードする関数を呼び出す
画像処理関数を呼び出す
画像ファイルをセーブする関数を呼び出す
バッファを解放する

img_io.c
画像ファイルのロード関数の定義
画像ファイルのセーブ関数の定義
画素値用バッファを確保する関数の定義
画素値用バッファを解放する関数の定義

img_proc.c
画像処理のための関数の定義

作業課題の目安: 10分