

プログラムテンプレートの ダウンロードと実行

1

【作業課題】ひな形のダウンロードと実行

- 課題4, 1週目の講義資料webページからプログラムテンプレートと画像ファイルをダウンロード
 - プロジェクトフォルダ"ex4"を作成し, main.cとimg01.ppmをダウンロードして同じフォルダにおく
 - img01.ppmをIrfanViewで表示できるか確認しておく
- 今週の目標は, 入力画像img01.ppmのヘッダ情報をresult.txtに出力させること
 - 実行してもまだresult.txtは作成されないことを確認

2

実行結果の表示について

- 「xxxx: No such file or directory」
 - 入力ファイルが存在しない(または入力ファイル名が間違っている)ため, ファイルの有無を要確認
- 「コード1で終了しました」(コードが0以外で終了)
 - 異常終了なので見直しが必要
 - 正常終了であればmain.cはコード0で終了する

3

作業課題の目安: 10分

4

プログラムテンプレートの流れ

プログラムの流れ

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

画素値を格納するための構造体
(詳細は2週目)

```
typedef struct {
    int    xsize;
    int    ysize;
    int    level;
    PIXEL  pBuffer[49152];
} IMAGE;
```

ヘッダ情報と画素値をまとめて
扱うための構造体で、
1週目はxsize, ysize, levelのみ使用

5

6

プログラムの流れ

プログラムの流れ

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

```
typedef struct {
    int    xsize;
    int    ysize;
    int    level;
    PIXEL  pBuffer[49152];
} IMAGE;
```

typedef structで定義しているので、
プログラム中で変数を宣言するときは
structを記載する必要なし

```
int main(...) {
    struct IMAGE a, b;
    ...
}
```

NG

```
int main(...) {
    IMAGE a, b;
    ...
}
```

OK

```
int main(int argc, char* argv[])
{
    IMAGE in_image_data, out_image_data;

    /* 画像情報のロード */
    if (iioLoadFile(&in_image_data, "img01.ppm")) {
        return 1;
    }
    /* 画像情報のコピー */
    ipCopy(&in_image_data, &out_image_data);
    /* 画像情報の書き出し */
    iioSaveFile(&out_image_data, "header.txt");
    /* メモリリークチェック */
    _CrtDumpMemoryLeaks();
    return 0;
}
```

構造体の変数の宣言

画像ファイルからヘッダ情報を
in_image_dataに
読み込む関数

in_image_dataの画像情報を
out_image_dataにコピーする関数

out_image_dataをテキスト
ファイルに出力する関数

7

8

プログラムの流れ

```
void ipCopy(IMAGE* p_in_image, IMAGE* p_out_image)
{
    /* ヘッダ情報のコピー */
    p_out_image->xsize = p_in_image->xsize;
    p_out_image->ysize = p_in_image->ysize;
    p_out_image->level = p_in_image->level;
}
```

入力ファイルから読み込んだ画像情報のうち、ヘッダ情報（横画素数、縦画素数、階調数）のみをp_out_imageにコピーしている。（画素値のコピーは2週目に行うので、1週目は省略）

main関数内のIMAGE構造体を編集する必要があるため、ipCopy関数へはポインタ渡しを行っている点に注意

（補足）ポインタ渡し（アドレス渡し）

■ 呼出し元の変数を呼出し先の関数内で変更可

■ 値渡しだと変更できない

値渡しの例

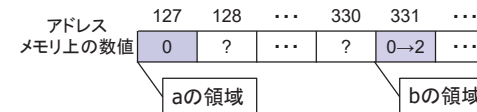
```
int main(void)
{
    int a = 0;

    func(a);
    printf("%d¥n", a);

    return 0;
}

void func(int b)
{
    b = 2;
}
```

➡ 0が出力される



ポインタ渡し（アドレス渡し）の例

```
int main(void)
{
    int a = 0;

    func(&a);
    printf("%d¥n", a);

    return 0;
}

void func(int *b)
{
    *b = 2;
}
```

➡ 2が出力される

