f_prac01.c

テキストファイル(score.txt)に

名前と**スコア**を記録する

①名前は最初にキーボードから入力する
②そのあと、何かキー入力するたびにスコアが＋1される
③アルファベットの「e」キーが押されたら入力を終了
④名前とスコアの値をファイルscore.txtに書き込む
⑤score.txtから読み込んで、画面に
　〇点:名前
　と表示する

f_prac01.c

実行結果

```
プレイヤーの名前＞神戸電子
現在のスコア:0 (‘e'で終了)
現在のスコア:1 (‘e'で終了)
現在のスコア:2 (‘e'で終了)
現在のスコア:3 (‘e'で終了)

e  ←eの入力で繰り返し終了

3点:神戸電子
```

↓score.txt

```
3, ○○  ←名前
↑スコア
```

f_prac01.c

～処理の順番～

①プレイヤーの名前を入力

②**無限ループ**の中で‘e’以外のキーが入力されたら

スコアを加算(‘e’が入力されたら強制終了)

③score.txtに名前と最終的なスコアを**書き込む**

④score.txtから名前と最終的なスコアを**読み込む**

f_prac01.c

使用する変数や配列～

```
//変数・配列の宣言
int score = 0;   //スコア
char ch;         //入力したキー
char name[20];   //名前を格納する配列
FILE* fp;        //ファイルポインタ
```

f_prac01.c

キー入力待ちは**getch()**で行う

実行したときに**入力待ち状態**になる

何らかのキー入力があれば、入力された

キーの**文字コードを戻り値**として返す

f_prac01.c

```
#include<stdio.h>
main()                          //f_prac01.c
{
  int score = 0;                //  スコア格納用変数
  char ch, name[20];            //  文字型変数および文字型配列
  FILE* fp;                     //  ファイルポインタ
```

f_prac01.c

```
printf("プレイヤーの名前は:");
scanf(“%s”, name);      // プレイヤー名の入力
while (1) {             // 無限ループ
  printf("現在のスコア:%d
          (Enterでスコア+1,eで終了)¥n",score);
  ch = getch();         // キーボードから一文字分入力
  if (ch == ‘e’) {      // 入力されたキーが’e’のとき
    break;              // 無限ループから抜ける
  }
  score++;          // ‘e’でなければscoreを+1してループ
}
```

f_prac01.c

```
// score.txtを書き込みモード(新規作成)して開く
if (fp = fopen("score.txt", "w")) {
  // ファイルにプレイヤー名とスコアを書き込む
  fprintf(fp, "%d,%s¥n", score, name);
  fclose(fp);    // ファイルを閉じる
}
else {  // score.txtの作成時にエラーが出た場合
  printf("エラー:ファイルを開けません¥n");
}
```

f_prac01.c

```
  // score.txtを読み取りモードで開く
  if (fp = fopen(“score.txt”, “r”)) {
    // ファイルからスコアとプレイヤー名を読み込む
    fscanf(fp, “%d,%s¥n”, &score, name);
    fclose(fp);    // ファイルを閉じる
    printf(“%d点：%s¥n”, score, name);
  }
  else {  // score.txtを開くときにエラーが出た場合
    printf(“エラー:ファイルを開けません¥n”);
  }
} //main()の終わり
```

f_prac02.c

↓ status.txt

```
隊員A,100,ショットガン,5,30.5
```

- 構造体のメンバにファイルから読み取ったステータスをセットし、実行結果に表示させる

実行結果

```
隊員A          体力:100
武器:ショットガン        残弾数:5        攻撃力:30.50
```

f_prac02.c

fscanfを用いた読み込みの注意点

```
fscanf(fp,"%s,%d,%s,%d,%f",･･･)
```

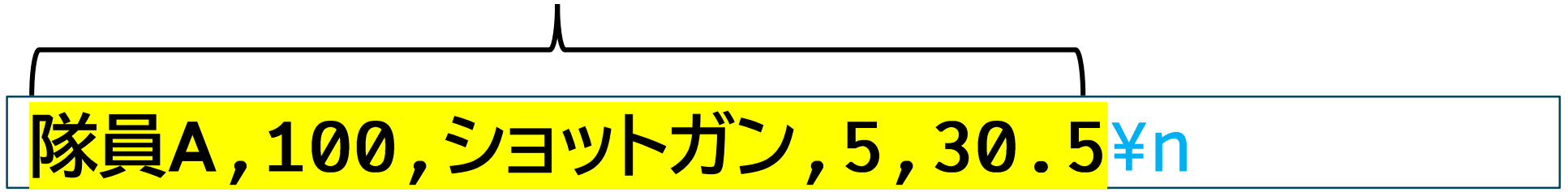
としてしまうとうまく読み込みができない

```
隊員A,100,ショットガン,5,30.5¥n
```

f_prac02.c

fscanfを用いた読み込みの注意点

```
fscanf(fp,"%s,%d,%s,%d,%f",･･･)
```



%sの変換指定子が、改行までの全文を
ひとつの文字列として読み込んでしまう…

f_prac02.c

fscanfを用いた読み込みの注意点

```
fscanf(fp,"%[^,],%d,%[^,],%d,%f",
```

```
隊員A,100,ショットガン,5,30.5¥n
```

%[^文字]を使うと、文字を除いた範囲を
文字列として読み込んでくれる！

f_prac03.c

- マップ情報を持ったテキストファイル**3つ**作成し、

  選択したマップを読み込み表示する

map0.txt

```
1111111111
1000000001
1000000001
1000000001
1111111111
```

map1.txt

```
1111111111
1000000001
1111111111
1000000001
1111111111
```

map2.txt

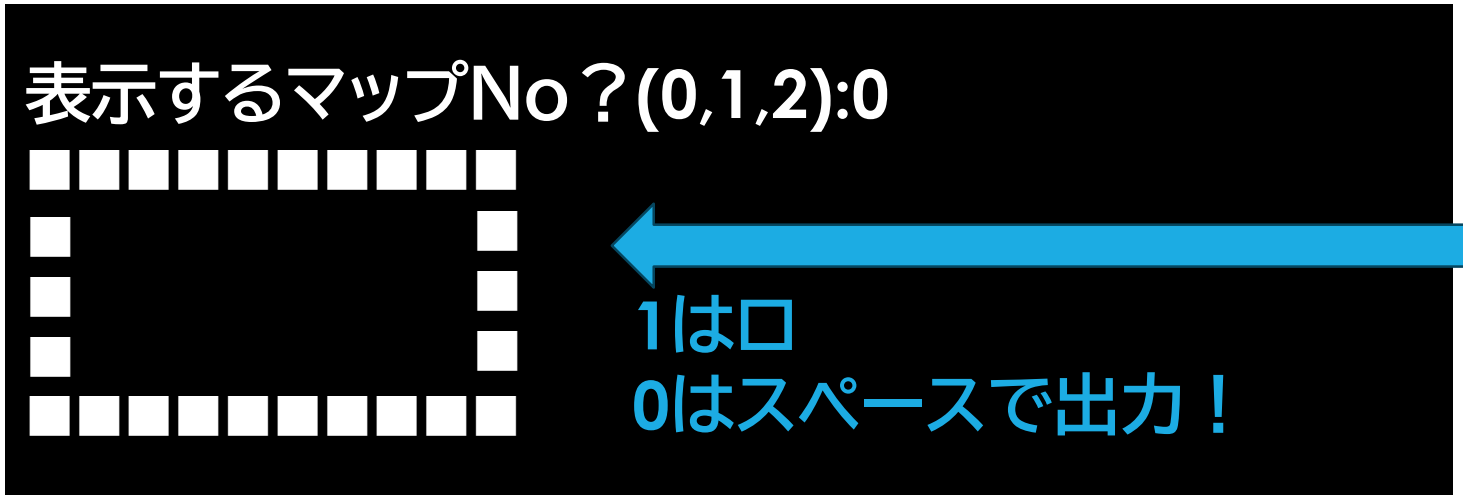
```
1111111111
1111111111
1111111111
1111111111
1111111111
```

f_prac03.cと同じフォルダに
マップデータを書き込んだ
テキストファイルを3つ用意

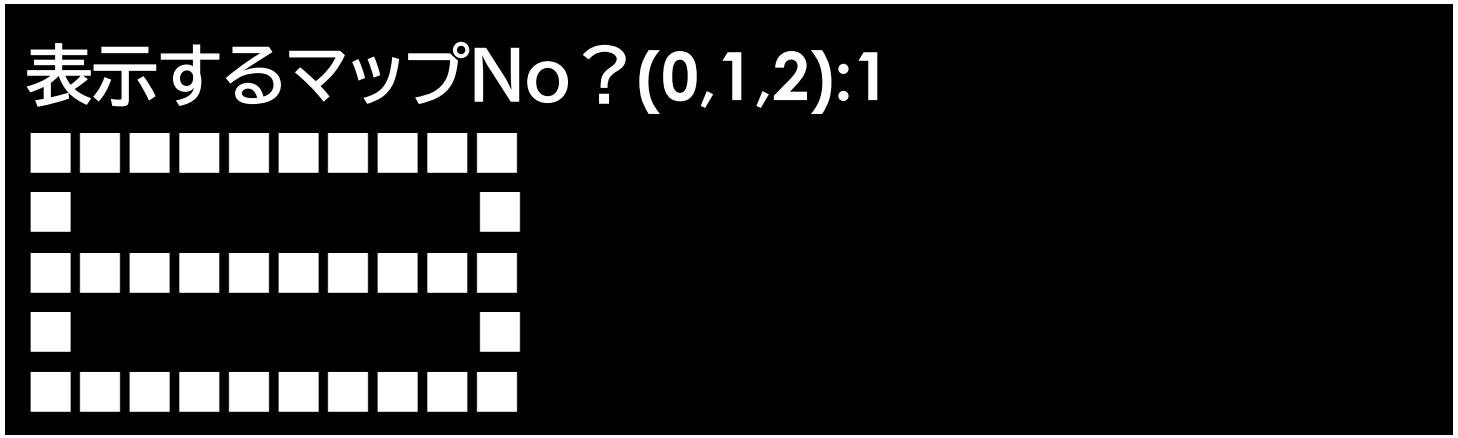
横10 縦5は必ず守ること！

f_prac03.c

実行結果 1



実行結果 2



map0.txt

```
1111111111
1000000001
1000000001
1000000001
1000000001
1111111111
```

map1.txt

```
1111111111
1000000001
1111111111
1000000001
1000000001
1111111111
```

f_prac03.c

～宣言するマクロ～

```
#define   MapNum   3  //マップの数
#define   W 10         //マップの横(10マス)
#define   H 5          //マップの縦(5マス)
```

～宣言する構造体(**typedef**を使用)～

```
typedef struct
{
  int  m_map[H][W];
} Map;
```

f_prac03.c

～宣言する変数～

```
Map MapData //マップのデータを管理する構造体;

//マップデータの配置を書いたテキストファイルの名前
char* MapFileName[MapNum]
= { "map0.txt","map1.txt","map2.txt" };

int select; //表示するマップの番号入力
```

f_prac03.c

～宣言する関数～

```
//選択したマップデータを配列にセットする関数
void setMap(char *filename,Map*m);

//マップデータ表示する関数
void drawMap(Map m);
```

※どちらも**main関数**の中で宣言します

f_prac03.c

～ SetMapの関数定義について～

```
//選択したマップデータを配列にセットする関数
void SetMap(char *filename,Map *m);
```

第一引数は選択されたマップの**ファイル名**([0]～[2]の指定)

第二引数は**Map構造体変数**の**アドレス**

☆やること☆

第一引数で渡されたファイルを**”読み取りモード”**で開き

第二引数 **m** のメンバである**二次元配列にセット**する！！

f_prac03.c

～ SetMapの関数定義について～

fgetcを使ってテキストファイルから**1文字ずつ**読み取っていく

<書式>

```
char ch = fgetc(ファイルポインタ)
```

戻り値:　①**読み取った1文字**

②**ファイル端まで読み込んだら**EOFを返す

f_prac03.c

～ setMapの関数定義について～

▶読み取りの流れ

①**fgetc**でファイルの終端まで一文字読み取り

char型の変数**ch**で受け取る

②数字を文字として読み取っているので**整数に変換**する

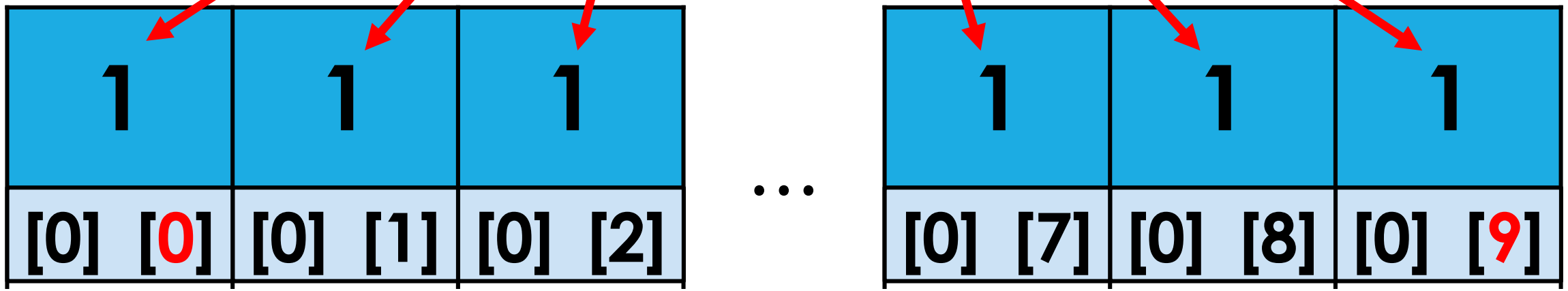
③変換後の整数を**二次元配列**にセットしていく

一行目→

map0.txtの

1111111111

1

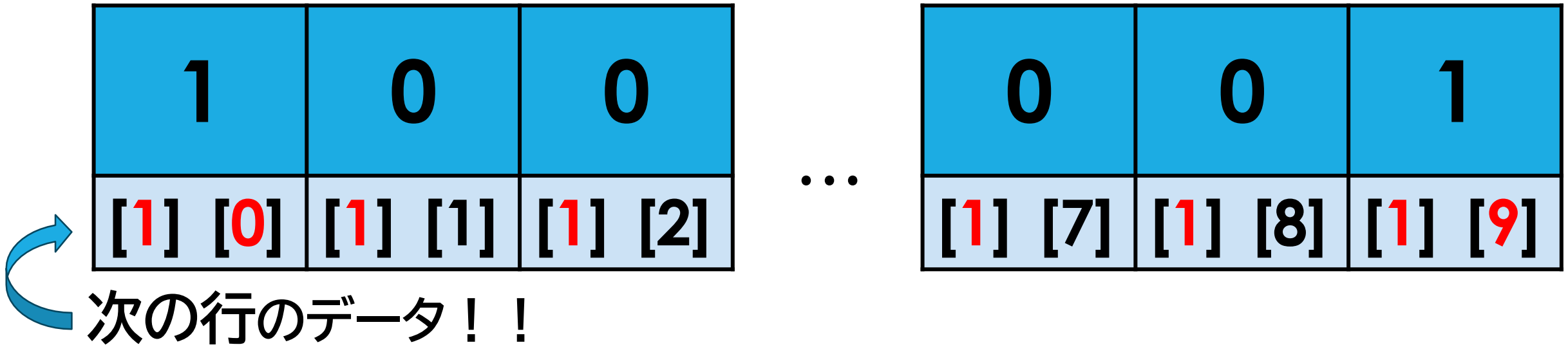


先頭データ

横10マス目

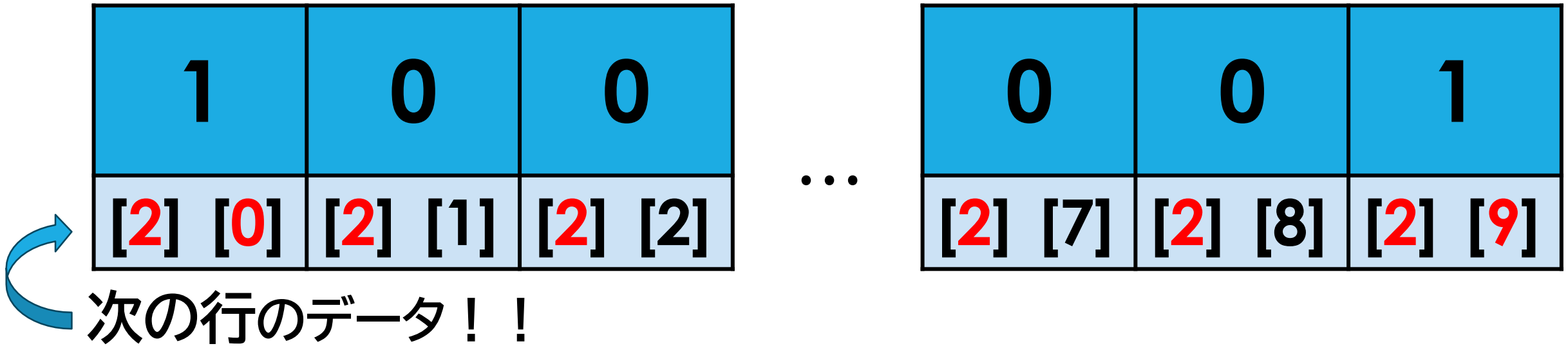
map0.txtの
二行目→

1111111111
1000000001



map0.txt
の三行目
→

1111111111
1000000001
1000000001



f_prac03.c

※注意※

二次元配列には**0**か**1**のデータのみ格納する

‘¥n’が入らないように処理を考えてみる

1111111111¥n
1000000001¥n ←見えないけど行の末尾に¥nがついてる
1000000001¥n
1000000001¥n
1111111111¥n

map0.txt

f_prac03.c

～ drawMapの関数定義について～

```
//マップデータ表示する関数
void drawMap(Map m);
```

第一引数は表示するマップの情報を持った**Map構造体変数**

関数機能

第一引数で渡された**構造体変数**のメンバの二次元配列から
1なら□、0なら空白で表示する！