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
 - ○点:名前
 - と表示する

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

- ~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<u>書き込む</u>
- ④score.txtから名前と最終的なスコアを<u>読み込む</u>

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
```

キー入力待ちはgetch()で行う 実行したときに<u>入力待ち状態</u>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

```
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
         // 無限ループから抜ける
   break;
 score++; // 'e'でなければscoreを+1してループ
```

```
// 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");
```

```
// 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
```

↓ status.txt

▶ ファイルから読み取った ステータスを変数や配列にセットし、 画面上に表示する 隊員A,100,ショットガン,5,30.5

実行結果

隊員A 体力:100 武器:ショットガン 残弾数:5

攻擊力:30.50

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

fscanfを用いた読み込みの注意点

fscanf(fp, "%s, %d, %s, %d, %f", ···)

隊員A,100,ショットガン,5,30.5¥n

%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...

fscanfを用いた読み込みの注意点

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

<mark>隊員A</mark>,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!

マップ情報を持った<u>テキストファイル3つ</u>作成し、

選択したマップを読み込み表示する

map0.txt

map1.txt

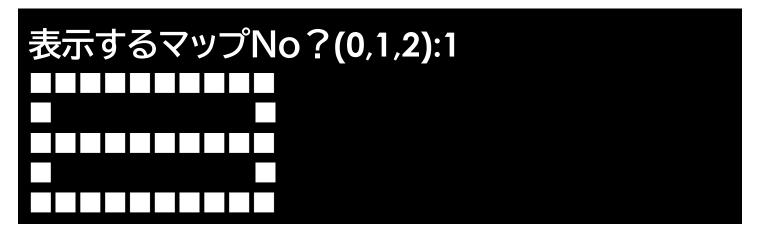
map2.txt

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!

実行結果 1

実行結果 2



map0.txt

map1.txt

~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;
```

~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
```

~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);
```

~ 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

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!!

~ 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

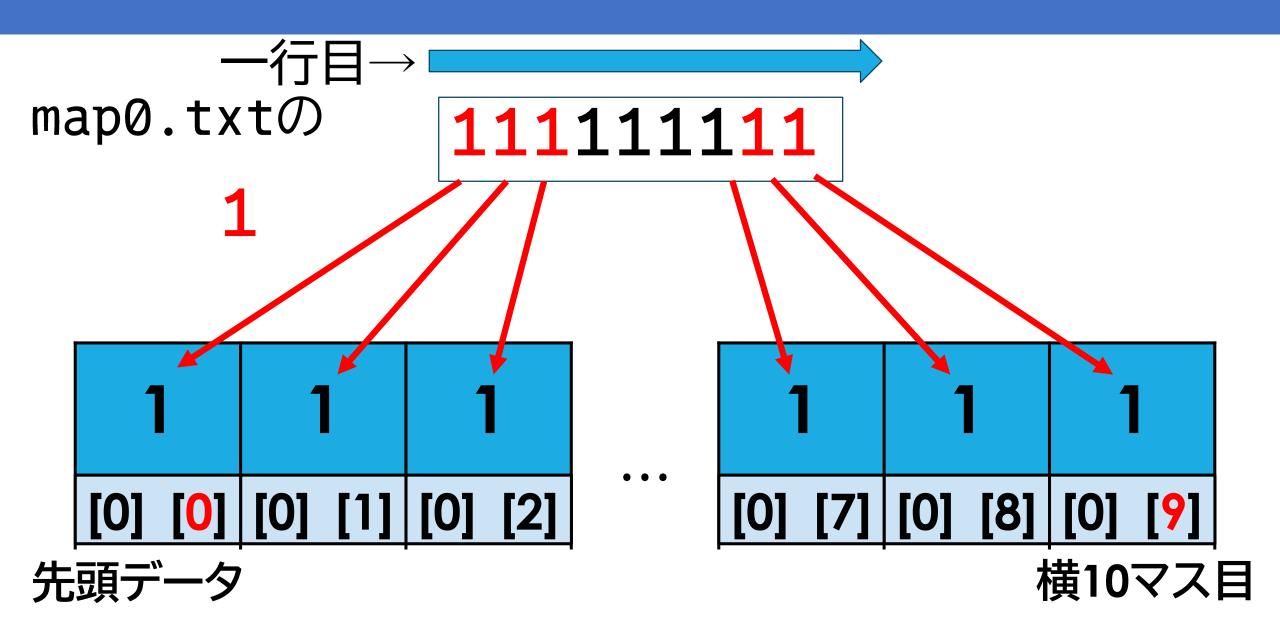
<書式>

char ch = $fgetc(7r4)\pi^{2}$

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

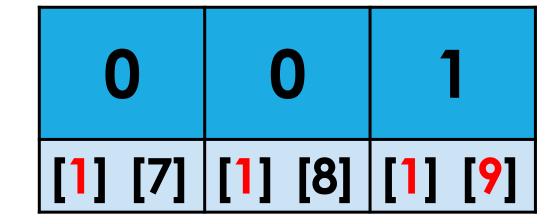
- ~ setMapの関数定義について~
 - ▶読み取りの流れ
- ①fgetcで行末まで一文字ずつ読み取っていく
 - → **char**型の変数**ch**で受け取る
- ②数字を文字として読み取っているので整数に変換する
- ③変換後の整数を二次元配列にセットしていく



map0.txt0 二行目→ 100000001

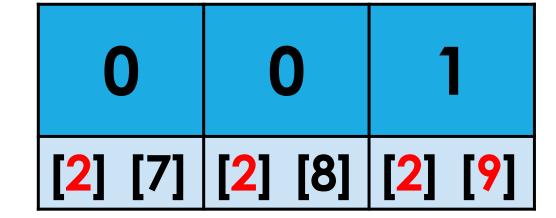
1111111111





map0.txt の三行目 11111111111111100000001 1000000001





※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

~ 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ならロ、0なら空白で表示する!