

プログラミング基礎

<http://bit.ly/prog2d>

ファイル入出力 (1)

ファイル入出力の基本

後期 第7週

2017/11/20

「入力」と「出力」 (p.388)

プログラムとプログラムの外部との間で
やり取りするデータの流を**ストリーム**と呼びます。

プログラム

```
#include <stdio.h>
int main(void)
{
    int x;
    FILE fp;
    printf("Hello!");
    scanf("%d", &x);

    fprintf(fp, "Hello!");
    fscanf(fp, "%d", &x);

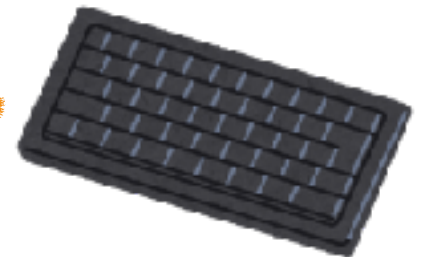
    return 0;
}
```

ストリームには出力 → と
入力 ← がある

ディスプレイ（標準出力）へ**出力**



キーボード（標準入力）から**入力**



ファイルへ**出力**
(書込み, セーブ)



ファイルから**入力**
(読込み, ロード)

ファイルへの読み書きも、ディスプレイ・キーボード
と同様に**ストリーム**の入出力として扱う

ファイル入出力の手順（p.401）

ファイルに対する読込み（入力）、書込み（出力）は、
大まかに**3つのステップ**からなります。

- ① ファイル**開く**（オープンする）
- ② **読み/書き**（入力/出力）する
- ③ ファイルを**閉じる**（クローズする）

① ファイルを開く (p.403)

「ファイルを開く」には、**fopen()** という関数を使います。

オープンしたファイルを参照する
ポインタ (**ファイルポインタ**)

【例1】

```
FILE *fp1, *fp2, *fp3;  
fp1 = fopen("test1.txt", "w");  
fp2 = fopen("test2.txt", "r");  
fp3 = fopen("test3.txt", "a");
```

オープンするファイル名

戻り値として、ファイルを参照するポインタ
(**型はFILE ***) が戻ってくる。**ファイルが開け
なかった場合はNULL**が戻ってくる。

オープンモード
w ... 書き込み
r ... 読み込み
a ... 追記
(p. 404 表12-3)

③ ファイルを閉じる (p.404)

「ファイルを閉じる」には、**fclose()**という関数を使います。

【例2】

```
fclose(fp1); /* ファイルtest1.txtの書きみを閉じる */  
fclose(fp2); /* ファイルtest2.txtの読みみを閉じる */  
fclose(fp3); /* ファイルtest3.txtの追記を閉じる */
```

クローズするファイルを参照している**ファイルポインタ**を指定する

オープンしたファイルの処理が終わったら**必ずクローズ**する

② ファイルに書き込む（出力する）

主に以下の3つの関数を使います。

fputs() … 文字列を1行出力する (p.405)

```
fputs("Hello,\n", fp1); /* fp1が参照しているファイルに"Hello,\n"を書き込む */  
fputs("Wrold.\n", fp1); /* 同じファイルの続きに"World.\n"を書き込む */
```

fprintf() … 文字列を書式つきで出力する (p.407)

```
fprintf(fp1, "No.%-5d%d\n", j+1, test[j]); /* p.407 Sample10.c より抜粋 */  
/* fp1が参照しているファイルに、書式で指定した文字列を書き込む(書式はp.390～396を参照) */
```

fputc() … ファイルに文字を1文字出力する (p.464)

```
fputc('A', fp1); /* fp1が参照しているファイルに、'A' を書き込む */  
fputc('z', fp1); /* 同じファイルの続きに、'z' を書き込む */
```


② ファイルから読み込む（入力する）

主に以下の3つの関数を使います。

fgets() … ファイルから1行入力する(p.409)

```
char str1[NUM], str2[NUM]; /* NUM は 20 とdefineされているとする */
fgets(str1, NUM-1, fp2);    /* fp2が参照しているファイルから、
                             最大NUM-1文字分を1行読み込みstr1に格納する */
fgets(str2, NUM-1, fp2);    /* fp2が参照しているファイルの続きから、
                             最大NUM-1文字分を1行読み込みstr2に格納する */
```

fscanf() … ファイルから書式つきで入力する (p.411)

```
fscanf(fp2, "%d", &test[i]); /* p.411 Sample12.c より抜粋 */
/* fp2が参照しているファイルから、書式で指定した文字列を読み込む */
```

fgetc() … ファイルから1文字入力する (p.464)

```
char ch;
ch = fgetc(fp2); /* fp2が参照しているファイルから、1文字読み込みchに格納する */
```

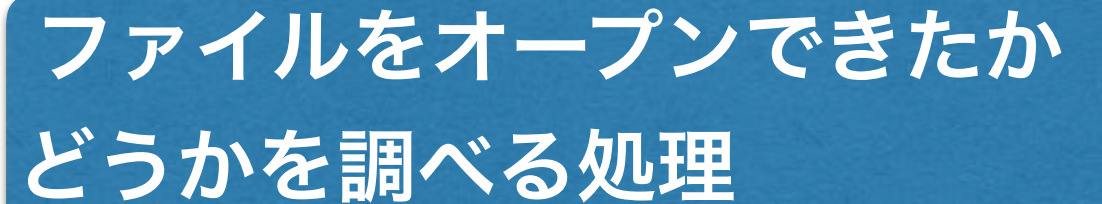
【練習7-1】

p.405 Sample9.c（または、次ページのプログラム）を入力して、ファイルへの出力処理を確認してみましょう。実行後にファイル「myfile.txt」を開いて、書き込まれた内容を確認してみましょう。

出力先のファイル名が既に存在する場合は、**上書きされてしまうことに注意してください。**


```
1: #include <stdio.h>
2:
3: int main(void)
4: {
5:     FILE *fp;
6:     fp = fopen("myfile.txt", "w");
7:     if(fp == NULL) {
8:         printf("Can not open the file.");
9:         return 1;
10:    } else {
11:        printf("The file is opened.\n");
12:    }
13:
14:    fputs("Hello.\n", fp);
15:    fputs("Goodbye.\n", fp);
16:
17:    fclose(fp);
18:    printf("The file is closed.\n");
19:
20:    return 0;
21: }
```

ファイルをオープンできたか
どうかを調べる処理



【練習7-2】

p.409 Sample11.c（または、以降のプログラム）を入力して、ファイルからの入力処理の様子を確認してみましょう。練習7-1で書き込んだ内容が、読み込まれて、画面に表示されることを確認してみましょう。

```
1: #include <stdio.h>
2: #define NUM 20
3:
4: int main(void)
5: {
6:     FILE *fp;
7:     char str1[NUM], str2[NUM];
8:
```

プログラムは次のスライドに続く

```
9:      fp = fopen("myfile.txt", "r");
10:      if(fp == NULL) {
11:          printf("Can not open the file.");
12:          return 1;
13:      } else {
14:          printf("The file is opened.\n");
15:      }
16:
17:      fgets(str1, NUM-1, fp);
18:      fgets(str2, NUM-1, fp);
19:
20:      fclose(fp);
21:      printf("The file is closed.\n");
22:
23:      printf("str1: %s\n", str1);
24:      printf("str2: %s\n", str2);
25:
26:      return 0;
27: }
```

【課題7-1】

練習7-1で入力したプログラムに対して、「scanf()によってキーボード（標準入力）から入力した文字列を、fputs()でファイルへ出力する」という処理になるように変更してください。

練習7-1からの変更点

- ▶ 文字列を格納するための配列を2個用意する
- ▶ scanf()によって入力した文字列を配列に格納する（2個分）
- ▶ fputs()で出力する文字列を、配列に置き換える（2箇所）

【課題7-2】

練習7-2で入力したプログラムに対して、NUMよりも長い文字列（つまり**20文字以上の文字列**）をファイルから読み込むと、str1とstr2にはどのような文字列が格納されているのか確認してください。

課題7-1のプログラムで20文字以上の文字列を入力した後に、練習7-2のプログラムを実行すると確認できます。

まだ余裕のある人は…

【課題7-3】

課題7-1で作成したプログラムに対して、開いたファイルに追加で書き込むように変更してください。

(オープンモード「w」はfopen()するたびに上書きされるので、モードを「追記」に変更します。)

まだまだ余裕のある人は…【課題7-4】

p.407 Sample10.cのプログラムに対して、点数を
小数点に対応できるように変更してください。

(配列testの型を、floatかdouble型に変更して、
fprintf()の出力書式も変更します。)