

C言語講座

～システム標準関数～

システム標準関数とは

システムに元々用意されている関数。

(必要なヘッダーファイルをincludeすることにより使用できる)

関数には大きく分けて、

- ・システム標準関数(入出力・その他)
 - ・自作関数
- があります。

入出力関数

- 標準出力関数 `printf(~);`
- 標準入力関数 `scanf(~);`

```
#include<stdio.h>
```

(ヘッダーファイルをinclude)

で使用できるのです。

ファイル入出力

様々なファイルにプログラムからデータを書き込んだり、そのファイルからデータを持ってきて利用したり。

ここでは、主にテキストファイルを使って基本的な動作を説明します！

- ・ファイルを開く `fopen`関数
- ・ファイルを閉じる `fclose`関数

fopen関数

```
FILE* fopen(char* file, char* mode);  
           ファイル名   ファイルを開くモード
```

↑
ファイル構造体のポインタ型(気にしなくてよい)

- mode
- “r” 読み出し。指定したファイルがないときは失敗
 - “w” 書き込み。指定したファイルがあるときは上書き
ないときは新規作成。
 - “a” 追加書き込み。
指定したファイルがあるときは追加。
指定したファイルがないときは新規作成。
 - “b” バイナリモード(“rb”, “wb”, …)

fclose関数

```
int fclose(FILE* fp);
```

オープンで指定したファイル構造体

ファイル入出力の流れ...

1. ファイルオープン
2. 読み書き
3. ファイルクローズ

ファイルを開いて(作って) 閉じるだけのプログラム

```
/* file1.c (例えば) */
```

```
#include<stdin.h>  
#include<stdlib.h>
```

```
Int main(void)
```

```
{
```

```
    char  fi[50] = "test.txt";
```

```
    FILE* fp;
```

```
    if ((fp = fopen ( fi, "w")) == NULL){
```

```
        fprintf(stderr, "Error: file open [%s].¥n", fi);
```

```
        exit (1);
```

```
    }
```

```
    printf("ファイルを開きました。¥n");
```

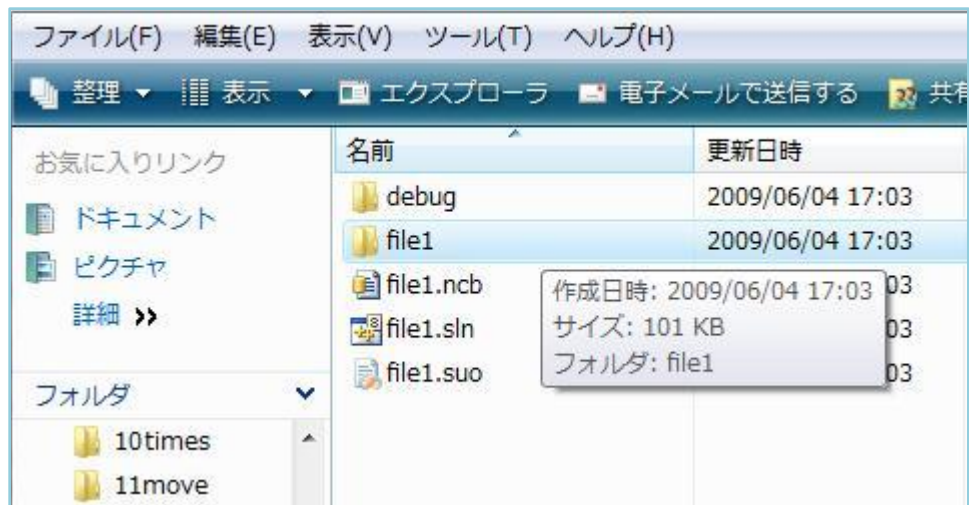
```
    fclose(fp);
```

```
}
```

← オープン

標準エラー出力

← クローズ



この中にファイルが作られる！
(test.txt)
何も書き込まれていない
ファイル

1文字入出力

- ・1文字入力

```
int getc(FILE* fp);
```

- ・1文字出力

```
int putc(int c, FILE* fp);
```

書き込む文字コード

オープンで指定したファイル構造体

- ・ファイルから1文字ずつ読み込んで画面に表示する

```
while((c = getc(fp)) != EOF)  
{  
    putc(c, stdout);  
}
```

End Of File (ファイルの最後)
まで繰り返す

標準出力

(ファイルではなく画面に表示)

練習問題

キーボードからファイルの名前を入力し、そのファイルの内容を1文字ずつ読み込んで画面に表示するプログラムを作成してください。(file2.c …例えば)

ファイル名に「file2.c」と入力して実行してみましょう！

～ヒント～

入力: `scanf("%s" , fi);`

画面に表示: 前のページ参照

オープンモードに注意！

乱数の生成

ゲームとか作る時には必要になると思います。

- 必要なインクルードファイル

```
#include<stdlib.h>    (乱数関数)
```

```
#include<time.h>     (時間関数)
```

- 乱数関数 `rand();`

0~RAND_MAXの整数の乱数

(RAND_MAX = 32767(通常))

しかしこのままではパターンが一定(らしい)なので、
変える必要がある。

時間関数の利用

- `srand()`;
 - 乱数の初期化
(引数を変えることで
乱数のパターンが変わる(?))

- 簡単な乱数生成

```
srand((unsigned)time(NULL));
```

乱数の精度は分かりませんが(えw
簡単に乱数を作るには便利。

乱数を発生させるプログラム(rand06.c)

```
#include<stdin.h>
#include<stdlib.h>
#include<time.h>

Int main(void)
{
    int i ;

    srand( (unsigned)time(NULL) );
    for(i=0;i < 30; i++){
        printf("乱数=%d", rand() % 100);
    }
}
```

0~99の乱数

ただ、
もっと良い乱数の作り方があるかも(すみませんw)

メモリの動的確保

普通、ユーザに数字のデータや文字列を入力させる場合、配列を
a[50];

などと余裕を持って宣言しますが、ユーザが3個のデータしか入力しない場合はメモリが無駄になってしまいますし、50個以上のデータや文字列を入力してしまうと、非常に危険です。

そのため、**必要な時に必要なだけメモリを確保する**のが望ましいです。。。

これをメモリの動的確保といいます。

(Javaではいつでもメモリが確保できますが・・・)

malloc関数

プログラムの実行中にメモリを確保する。

- ・必要なインクルードファイル

```
#include<malloc.h>
```

```
void *malloc( size_t size );
```

確保するbyte数

どの型にもキャスト(型変換)できるという意味

free関数 free(void *memblock);

動的に確保したメモリは解放する。

メモリ動的確保の利用例(malloc06.c)

```
#include<stdio.h>
#include<malloc.h>
```

```
int main(void)
{
    int *pi, i, n;

    printf("入力する整数型データ数:");
    scanf("%d", &n);
    pi = (int *)malloc(sizeof(int) * n);
    for(i=0;i<n;i++){
        printf("%d個目:", i + 1);
        scanf("%d", pi + i);
    }
    printf("¥n");
    for(i=0;i<n;i++){
        printf("a[%d] = %d¥n", i, *(pi + i));
    }
    free(pi); /* メモリの解放 */
}
```

ポインタを利用する！

/* a[n]を確保*/

↑ Int型のbyte数を取得

← = a[i]

ちょっと難しいかもしれませんがね。
といいますか、配列とポインタの関係はまだやって
いませんでした。少しだけ...

int *a; int a[]; ※int型は4byte

配列要素の値			番地(アドレス)		
ポインタ		配列	ポインタ	配列	アドレス
*a	=	a[0]	a	&a[0]	100
*(a+1)	=	a[1]	(a+1)	&a[1]	104
*(a+2)	=	a[2]	(a+2)	&a[2]	108
*(a+3)	=	a[3]	(a+3)	&a[3]	112
▪		▪	▪	▪	▪
▪		▪	▪	▪	▪
▪		▪	▪	▪	▪

前ページのプログラムでは、ポインタを配列のように考えて使っています。

お疲れ様でした！これで2年生による C言語講座は終了となります。

今日の内容は応用の範囲なので、基本を身につけてからで大丈夫です。こんなことをやった、ということだけ覚えておいて下さいね～。

ぶっちゃけ授業聞いてるだけでは、深くは理解できないし、知識の定着を図るため、他の人に差をつけるために、自主学習をオススメします！（既に得意な人以外？）

- ・猫でもわかるC言語プログラミング
- ・やさしいC

etc...（他にもたくさんあります、情科の方はJavaを）
一冊通してやってみると、いろいろ分かると思いますよ！

分からないことがあれば、お近くの先輩まで（笑）

練習問題の答え

```
#include<stdio.h>
#include<stdlib.h>
Int main(void)
{
    int c;
    char fi[50];
    FILE *fp;

    printf("Input file name :");
    scanf("%s",fi);

    if((fp=fopen(fi,"r"))==NULL){
        fprintf(stderr,"Error:file open[%s].¥n",fi);
        exit(1);
    }
    while( (c = getc(fp) ) != EOF){
        putc(c, stdout);
    }
    fclose(fp);
}
```

6/11(木) 18:30～

3年生の先輩の
プレゼンテーション
があります