C言語講座
第6回

システム関数

今回は、C言語に備わっている便利な関数群について、有用だと思われるものをいくつかピックアップして教えていこうと思う

・・・ので、ライブラリ関数を扱うために必要な、“ヘッダをインクルードする”ということを先に学んでいこう。

ヘッダ(header)

その名の通り…と言っていいのか分からないが、たいていはソースコードの上部、つまり頭に書きたいプログラムを記述したファイル。

具体的には、

プロトタイプ宣言、構造体・共有体の定義、
enum定義、グローバル関数の宣言、
インライン関数の宣言・定義

などなど。

インクルード(include)

ヘッダファイルで宣言したものを取り込む。

include自体には“含む”という意味があり、まぁ、ここでの意味的には“読み込む”程度に思ってくれれば良い。

○構文

```
#include "ファイル名"
```

○プログラミング実践

変数を入れ替える

test.h

```
#include "stdio.h"
void swap(int *x, int *y);
void print_ab(int a, int b);
```

```
#include "test.h"
void main(void) {
    int  a = 3, b = 9;
    print_ab(a, b);
    swap(&a, &b);
    print_ab(a, b);
}
void swap(int* x, int* y) {
    int  tmp = *x;
    *x = *y;
    *y = tmp;
}
void print_ab(int a, int b) {
    printf("a=%d, b=%d¥n", a,b);
}
```

イメージ的には
こうなる

```
#include "stdio.h"
void swap(int *x, int *y);
void print_ab(int a, int b);
void main(void) {
   int  a = 3, b = 9;
   print_ab(a, b);
   swap(&a, &b);
   print_ab(a, b);
}
void swap(int* x, int* y) {
   int  tmp = *x;
   *x = *y;
   *y = tmp;
}
void print_ab(int a, int b) {
   printf("a=%d, b=%d¥n", a,b);
}
```

ちなみに。

“#”がつくキーワードはプリプロセッサと呼ぶ、コンパイラに対する特殊命令。

#include、#define、#ifndef、#if、#endifなど、いろんな種類がある。

stdio.h(Standard Input/OutPut)

標準入出力を行うための機能が盛り込まれている。代表的なものを紹介する。

入出力:

printf, scanf, gets ,puts

ファイルアクセス:

fopen, fclose

gets, puts (get/put String)

行単位の入出力

具体例

```
#include <stdio.h>

main()
{
        char s[256];
        gets(&s);
        puts(&s);
}
```

time.h(time)

時間操作。

時間取得：

time

time, localtime

time:歴時刻(システム時間)の取得。
　　現在の日付を取得できる。

localtime:time関数で取得した時間を、
　　tm構造体(次々ページ)へ変換する。

どうして変換する必要があるの？

timeで取得するものは、少し語弊はあるがint型だ。1970年1月1日を0として、そこからの秒数を表している。
(例：1971年1月1日：86400(…多分))
だから、次ページのように、分かりやすいように変換する。

○プログラミング実践

今日の日付を表示する

```
#include <time.h>
#include <stdio.h>
void main() {
        time_t now;
        struct tm t;
        time(&now);
        t = *localtime(&now);
        printf("%d年%d月%d日¥n",
                t.tm_year + 1900, t.tm_mon+1, t.tm_mday);
}
```

構造体なので、
structを最初に書く

tm構造体

構造体は、幾つかの変数をまとめたものだと思っていい。

```
struct tm {
        int  tm_sec;     // 秒(0-60)
        int  tm_min;     // 分(0-59)
        int  tm_hour;    // 時(0-23)
        int  tm_mday;    // 日(1-31)
        int  tm_mon;     // 月(0-11)
        int  tm_year;    // 年(年-1900)    110なら2010年のこと
        int  tm_wday;    // 曜日(0-6)
        int  tm_yday;    // 1月1日よりの通し日付
        int  tm_isdst;   // 夏時間調整フラグ
}
```

stdlib.h(Standard library)

一般ユーティリティ。

ランダム:

srand, rand

メモリ管理:

malloc,free

ファイルアクセス:

fopen, fclose

fopen(file open)

filenameで示すパスのファイルを指定したmodeで開く。

constは定数の意。
ここでは気にしなくても良い。

○構文

```
FILE  *fopen(const char *filename, const char *mode)
```

mode:

mode	read/write	シーク	ファイル存在時	ファイルが無かった
r	read	先頭	何もしない	エラーで返す
w	write	先頭	空ファイルにする	空で作成する
a	write	終末	何もしない	空で作成する
r+	read/write	先頭	何もしない	エラーで返す
w+	read/write	先頭	空ファイルにする	空で作成する
a+	read/write	終末	何もしない	空で作成する

fclose(file close)

ストリームに結合したファイルを解放する。オープンしたファイルは必ずこれを行わなければならない。

〇構文

```
int  fclose(File *stream);
```

○プログラミング実践

ファイルの中身を表示する

```
#include <stdio.h>
void main()   {
        FILE* pf;
        char s[256];
        if ((pf = fopen("text.txt", "r")) != NULL) {
                fgets(&s, sizeof(s), pf);
                printf("%s", s);
                fclose(pf);
        } else {
                printf("file not found...");
        }
}
```

srand, rand (seed random, random)

srand:乱数の初期化

rand:乱数の取得(0 ~ 3)

具体例

```
#include<stdio.h>
#include<stdlib.h>
#include<time.h>
void main(void) {
	inti;
	srand( (unsigned)time(NULL) );
	for(i=0;i < 10; i++){
		printf("乱数＝%d¥n", rand() % 100);
	}
}
```

動的メモリ領域割り当てってって？

これの反対語が静的(ry)。

例えば、char s[1000][100];
という配列を宣言すると、1000*100=約100K byteのメモリを取得するが、これは実行時に変更することが出来ない。足りなくなるかも知れないし、多すぎて無駄かも知れない。

だから、

必要な時に必要なだけメモリを確保したい

malloc, free (メモリ管理)

malloc:指定バイト分、メモリ領域を確保する

free:メモリ領域を解放する

具体例

```
#include <stdlib.h>

void main()
{
        char *p;
        p = (char*)malloc(10);
        free(p);
}
```

char * p[10];
と同じ

○演習

今日の日付(年、月、日、曜日)をテキストファイルとして出力するプログラムを作成してください。

最後なんで、面倒なプログラムです。

文字列操作に関してはさらに難しいので自作関数を次ページに用意しました。

(自分でやれる人は見なくてもいいけどw)

次ページにあるプログラムのmain関数の中を補って完成させてください。必要に応じて関数を増やしても良い。

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include <locale.h>

void get_str_day(char* s, struct tm t) {
        setlocale(LC_ALL, "");
        strftime(s, 21, "%Y年%m月%d日%A", &t);
}
void main() {
        time_t now; struct tm t;
        char* s;     //動的に21文字分の領域を取得すること。
        FILE* pf;

        /*
        *  時間の取得
        */
        /*
        *  動的に取得
        */
        get_str_day(s, t);

        // ファイルへの書き込み

}
```

get_str_day(*s, 2010, 10, 10, 1)ならば、
sに”2010年10月10日 月曜日”と書きこまれる

○解答例(main関数の中身だけ)

```
void main() {
		time_t now; struct tm t;
		char* s;
		FILE* pf;

		time(&now);
		t = *localtime(&now);

		// 2010年月日日曜日など、最高でも文字だが、日本語は文字換算+¥0が必要なので+6+1
		s = (char*)malloc(sizeof(char) * 21);
		get_str_day(s, t);

		if ((pf = fopen("day.txt", "w")) != NULL) {
			fputs(s, pf);
			fclose(pf);
		} else {
			printf("file not created...");
		}

		free(s);
}
```