

プログラミング基礎

<http://bit.ly/prog2d>


標準ライブラリの利用

前期 第5週

2017/5/15

【Point 1】 標準ライブラリ関数が宣言されている
ヘッダファイルを読み込む (p. 266～267)

```
1:  /* 関数printf(), scanf() を  
2:     使うために読み込む */  
3:  #include <stdio.h>  
4:  
5:  /* 関数abs() を使うために読み込む */  
6:  #include <stdlib.h>  
7:  
8:  /* 関数isalpha() を使うために読み込む */  
9:  #include <ctype.h>  
10:  
11: /* 関数sin() を使うために読み込む */  
12: #include <math.h>  
13:
```



【Point 2】 どのヘッダファイルに、どのような標準ライブラリ関数が宣言されているのかは、テキストの付録Bを参照 (p. 464～467)

【Point 3】関数abs()は、引数で渡した整数値の絶対値を返す (p. 465)

```
14: int main(void)
15: {
16:     int num1 = -30;
17:     char ch;
18:
19:     printf("%d\n", abs(num1));
20:
21:     printf("%f\n", sin(1));
22:
```

【Point 4】関数sin()は、引数で渡した数値（単位はラジアン）を計算する (p. 466)

【Point 5】関数isalpha()は、引数がアルファベットならば真、アルファベットでないなら偽となる(p. 466)

```
23:    printf("ch > ");
24:    scanf("%c", &ch);
25:    if( isalpha(ch) ) {
26:        printf("%cはアルファベットです。\\n", ch);
27:    } else {
28:        printf("%cはアルファベットではありません。\\n",
                ch);
29:    }
30:
31:    return 0;
32: }
```

【Point 6】isで始まる関数は真偽を返す関数なので、if文の条件に使える

コンパイルとリンクについて

プログラムは以下のように、**コンパイル**と**リンク**の処理を経て実行可能な状態に変換されます。

「cc」はコマンド1つで、コンパイルとリンクの処理を行ってくれます。

ソースコードファイル (???.c) + ヘッダファイル (???.h)



コンパイル

オブジェクトファイル



リンク

実行可能なファイル (a.out)

標準ライブラリを使う場合…

標準ライブラリ関数を使ったプログラムをコンパイルする際は、**ライブラリファイルをリンクする必要があります。**

ライブラリをリンクするには、コンパイル時に「-l」の後ろに**ライブラリファイル名を指定します。**

今回紹介したライブラリの場合

- ❖ math.hを使う場合は、「-lm」を付ける
- ❖ ctype.hを使う場合は、「-lc」を付ける

(例： `$ cc myprog.c -lm -lc`)

【練習5-1】

サンプルプログラムをコンパイルして、
実行結果を確認しましょう。

【課題5-1】

関数main()にて、scanfで入力した整数xに対して「 \sqrt{x} 」と「 2^x 」を出力するプログラムを作成してください。使用する関数はp. 466の<math.h>を参照しましょう。（コンパイルには「-lm」を指定する）

[実行結果]

x > 3

(← “x > ” が出力され、続けて3を入力した)

sqrt: 1.732051

pow : 8.000000

【課題5-2】

グローバル変数として宣言されたint型の配列test
(要素数は5個) に対して、各要素の値を絶対値へと
変換する関数abs_array()を作成して下さい。

[この関数のプロトタイプ宣言]

```
void abs_array();
```

```
/* 配列testに対する繰り返し処理の中で、
```

```
    標準ライブラリ関数abs()を使うように作ってみる */
```

```
/* 処理の結果は配列testに反映される */
```

【課題5-2】

配列の出力には、以下の関数show() が必要です。
(第4週サンプルプログラムからコピー可能)

```
#define N 5

/* 配列testの要素を出力する */
void show()
{
    int i;
    for(i=0; i<N; i++) {
        printf("test[%d]: %d\n", i, test[i]);
    }
}
```

【課題5-2】

[配列testをグローバル変数として宣言する]

```
int test[5] = {-12, 25, -8, -32, 5};
```

[main() での処理]

```
abs_array();  
show();
```

[実行結果]

```
test[0]: 12  
test[1]: 25  
test[2]: 8  
test[3]: 32  
test[4]: 5
```

【課題5-3】

引数で受け取った文字をその種類に応じて変換し、
変換された文字を戻り値とする関数convert_char()
を作成してください。

[この関数のプロトタイプ宣言]

```
char convert_char(char c);
```

```
/* 以下のように仮引数cの文字の種類に応じて、戻す文字を決める */
```

```
/*      ・ cが小文字の場合、その大文字を戻す */
```

```
/*      ・ cが大文字の場合、その小文字を戻す */
```

```
/*      ・ cが数字の場合、 '*' を戻す */
```

```
/*      ・ それ以外は変換されず、cが持つ文字をそのまま戻す */
```

```
/* これらの処理には、ctype.hの標準ライブラリ関数を使う (p.466) */
```

【課題5-3】

[main()での処理]

```
printf("convert: %c\n", convert_char('B'));  
printf("convert: %c\n", convert_char('m'));  
printf("convert: %c\n", convert_char('5'));  
printf("convert: %c\n", convert_char('@'));
```

[実行結果]

```
convert: b  
convert: M  
convert: *  
convert: @
```

【課題5-4】

double型の配列test2に対して、要素の値に応じて
数値の繰り上げ / 繰り下げ処理をする関数
convert_array()を作成してください。

[この関数のプロトタイプ宣言]

```
void convert_array();
```

```
/* 配列test2に対する繰り返し処理の中で、
```

```
    以下のような条件に応じて処理を分ける */
```

```
/*      ・要素の値が0以上100未満の場合、その値を繰り上げる */
```

```
/*      ・要素の値が100以上200未満の場合、その値を繰り下げる */
```

```
/*      ・それ以外は変換されない */
```

```
/* これらの処理の結果は配列test2に反映される */
```

```
/* 繰り上げ/繰り下げには、math.hの標準ライブラリ関数
```

```
    ceil(), floor()を使う (p.467参照) */
```

【課題5-4】

配列test2の出力には別途関数が必要です。
(関数 show()を参考に作れる)

```
#define N 5

/* 配列test2の要素を出力する */
void show2()
{
    int i;
    for(i=0; i<N; i++) {
        printf("test2[%d]: %f\n", i, test2[i]);
    }
}
```

【課題5-4】

[配列test2をグローバル変数として宣言する]

```
double test2[5] = {10.2, 125.8, 382.5, 20.9, 107.1};
```

[mainでの処理]

```
convert_array();  
show2();
```

[実行結果]

```
test2[0]: 11.000000  
test2[1]: 125.000000  
test2[2]: 382.500000  
test2[3]: 21.000000  
test2[4]: 107.000000
```


まだ余裕のある人は…【課題5-5】

int型の配列testに対して、要素の値の平方根を求めて配列test2に格納する関数sqrt_array()を作成してください。ただし、**値が負の場合は、その絶対値の平方根が格納される**ようにしてください。

[この関数のプロトタイプ宣言]

```
void sqrt_array();
```

```
/* 配列testに対する繰り返し処理の中で、平方根を求め、  
   その結果をtest2に格納する */
```

```
/* 標準ライブラリ関数sqrt()とabs()を組み合わせるを使う */
```

```
/* 処理した結果は配列test2に格納され、配列testは変更されない */
```

【課題5-5】

[配列testをグローバル変数として宣言する]

```
int test[N] = {-2, 25, -9, -10, 5};
```

[mainでの処理]

```
sqrt_array();  
show2();
```

[実行結果]

```
test2[0]: 1.414214  
test2[1]: 5.000000  
test2[2]: 3.000000  
test2[3]: 3.162278  
test2[4]: 2.236068
```

小テストについて

小テストの注意点

- 他人の力は借りずに、自分だけでプログラムを作成する。（つまり定期試験と同様）
- 小テスト中は、演習室外へのネットワークアクセスは遮断される。

小テストについて

小テスト中に参照できるもの

- 教科書, 配付資料
- 自分のホームディレクトリ（ホームフォルダ）以下に保存されているファイル
- 小テストでは紙媒体のものは参照可能
- 上記以外の情報を参照することはカンニング行為とする
例：USBで接続された機器に保存されているファイルの参照
ネットワークを介した情報の参照、など