プリプロセッサーとコンソール入出力関数

山本昌志*

2006 年 6 月 19 日

概 要

プリプロセッサーのうち，本講義で重要なファイルの挿入`#include` と文字列置換`#define` について学習する． また，ディスプレイにデータを出力する方法と，キーボードからデータを読み込む方法を学習する．数値計算に必要な最小限のことを説明している．

1 本日の学習内容

本日は教科書 [1] の 15 章プリプロセッサと 16 章コンソール入出力関数について学習する．主にデータ構造に関わる 12〜14 章は省く．数値計算のデータ構造は単純なので，ほとんど場合，配列で間に合う．しかし，他の多くのアプリケーションでは，アルゴリズムと並んでデータ構造は重要である．いろいろなアプリケーションを作成したい者は，12 章〜14 章を自習せよ．複雑なデータ構造を使う必要がる場合は，12 章の構造体は必須である．また，PIC や H8 などのマイコンのプログラムを作成する場合，ビットフィールド (13 章) と共用体 (14 章) を理解しなくてはならない．

教科書の 15 章と 16 章で以下のことが分かればよい．

- 15 章 プリプロセッサーでは以下が理解できればよい．
  - プリプロセッサーは，コンパイル前に言語のソースプログラムの処理を行う．
  - `#include` は，ファイルのテキストを挿入する．
  - `#define` は，文字列の置き換えを行う．
- 16 章 コンソール入出力関数では以下のことができることを目指す．
  - `printf()` 関数を使って，任意の書式で数値をディスプレイに出力できる．
  - `scanf()` 関数を使って，キーボードからデータを入力できる．

2 プリプロセッサー

2.1 プリプロセッサの実行タイミング (p.282)

「gcc」を使って，ソースプログラムをマシン語に直すとき，通常はコンパイルすると言うが，実際にはコンパイル以外の作業も行っている．教科書の p.282 に示されているように，コンパイルの前にプリプロ

*独立行政法人　秋田工業高等専門学校　電気工学科

セッサーがソースファイルを書き直している．そして，コンパイルにより，ソースファイルをオブジェクトファイルに変換している．最後に，リンカーがいろいろなファイルを寄せ合わせて実行可能なファイルを作成する．

プリプロセッサーは，コンパイルに先だって，ソースファイルを変更している[1]．gcc ではいろいろな処理が行われるが，プリプロセッサーの動作が一番最初に実施される．そこで，諸君が書いた C 言語のプログラムを変更しているのである．preprocessor(pre:あらかじめ，processor:処理するもの) という名前からも，そのことがよくわかる．

プリプロセッサーができることは，教科書の p.283 の表に示されている．いろいろな機能があるが，本講義で使うのは

- ファイルの挿入を行う`#include`
- マクロ定義を行う`#define`

くらいである．

2.2 プリプロセッサの使い方 (p.282)

2.2.1 プリプロセッサの書き方

プリプロセッサーの書式は，以下の通りである．今まで，さんざん書いてきているので，大体理解はできるであろう．

```
#コマンド　パラメーターリスト
```

このプリプロセッサーの書き方には，以下の約束がある．

- #の前後に空白が有ってもよい．#とコマンドの間の空白も許されるが，プログラムがわかりにくくなる．通常は#とコマンドの間に空白を入れない．
- プリプロセッサーコマンドは，その行で終わりである．C 言語の文のように';' があらわれるまで有効ということではなく，その行で完結する．このようなことから，文の区切りを表す';' を書いてはならない．
- ただし，プリプロセッサーコマンドを複数行にわたって，記述したい場合，行の終わりに'\' をつけて，次行と接続することができる．

2.2.2 ファイルの挿入 (p.284)

`#include` の役割と例　`#include "ファイル名"`は，指定したテキストファイルとこの行を置き換える．ファイル名が`<filename>`と鈎括弧<>で書かれた場合には，システムに定義されたファイルを示す．unix では，`/usr/include` にあるファイルで置き換えられる．プログラマーが作成したファイルを挿入したい場合は，`#include "myfile.h"` のように記述する．

[1] 教科書に書いてあるとおり，プリプロセッサーには他にも役割がある．それについては，割愛する．

プログラマーが自分でヘッダーファイルを書くことがある．例えば，大規模なプログラムを作成する場合，ファイルは 1 つではなく，いろいろな部分を別々のファイルに書いて，それをあとであわせて，1 つのプログラムにする（分割コンパイル）．その場合には，共有する構造体や大域変数を 1 つのファイルにして，myfile.h というようなファイルを作り，それぞれのソースプログラムでインクルードする．このようにすると，同じ文をそれぞれのファイルに記述する必要が無くなり，タイピングが楽になるとともに，ミスが減る．ここでの学習では，分割コンパイルをするほどのプログラムを書くことはない．将来，比較的大きなプログラムを書くときに，勉強すればよいだろう．

#include の動作の理解のために，へんなプログラム例を示す．Hello World !!を出力するおなじみのプログラムである．

リスト 1: ソースプログラム

```
1 #include <stdio.h>
2 #include "hoge"
3  return 0;
4 }
```

これをコンパイルして，実行ファイルを作るためには，以下のファイル (hoge) が必要である．当然，このファイルはソースプログラムのインクルード文で取り込まれる．

リスト 2: インクルードするファイル (hoge)

```
1 int main(void){
2  printf("Hello World !!\n");
```

実行の結果，#include により，そこの行が指定のファイルに置き換わっているのが理解できるはずである．使い方によっては，便利そうであることが理解できればよい．この行の置き換えはコンパイルの前に行われることに注意が必要である．

ヘッダーファイル　諸君は，プログラムの最初に必ず#include <stdio.h>とパブロフの犬の如く書いていただろう．ここにきて，#include の意味が分かった．stdio.h というファイルをインクルードしているだけだ．そうなると，stdio.h を見たくなるのが人情というものである．幸い，これはテキストファイルなので見ることができる．以下のコマンドをターミナルへ打ち込んで，stdio.h を見よう．

```
less /usr/include/stdio.h
```

キーボーの f で前のページを，b で後ろのページを見ることができる．終わりたい場合は，q を押す．

その中をみると，関数のプロトタイプ宣言等がごちゃごちゃ書かれているのが分かるだろう．これが，諸君のプログラムの最初の方に追加されるのである．これで，printf() などの関数が使えるようになるのである．

2.2.3 マクロ定義 (p.287)

文字列置換　#define はファイル内の文字列を指定の通りに書き換える．単純な文字列の置き換えと，引数を含む文字列の置き換えがある．引数をとるマクロ定義は，本講義では使う必要がないので，ここでの学習範囲外とする．

単純な文字列置換をオブジェクト形式マクロと言う．これは，次のように書く

```
#define hogehoge fugafuga
```

このプリプロセッサーコマンドがあると，この行以降の文字列 hogehoge が文字列 fugafuga に，すべて置き換わる．

置き換え前の文字列—ここでは hogehoge—はすべて大文字で記述する習慣となっている．別に小文字でも問題なく動作するが，ほとんどのプログラマーは大文字を使う．その方が，プログラムがわかり易いからである．

プログラム例　リスト 3 に，文字列置換を使ったプログラムラム例を示す．4 行目の文字列置換

```
#define NSTEPS 361
```

により，全ての NSTEPS という文字列が，361 に変えられる．このようにすることにより，同じ値の部分を一度に変えることができる．プログラムミスが減るので，便利である．

リスト 3 のプログラムは，三角関数の数表を出力する．0〜2π ラジアンを 360 等分して，関数の値を計算，そして出力している．例えば，500 等分の値を出力したければ，#define NSTEPS 501 とする．文字列置換を使えば，プログラムの変更が容易であることがわかる．

このプログラムの 31 行目に，M_PI というわけの分からない変数らしきものがある．実は，これも文字列置換の対象である．ヘッダーファイル math.h の中に，

```
# define M_PI 3.14159265358979323846 /* pi */
```

と書かれている．ようするに，円周率 π の値に変換される．ウソだと思うならば，

```
less /usr/include/math.h
```

とコンソールに打ち込んで調べよ．

このプログラムは，数学関数—こでは三角関数—を使っている．そのため，ヘッダーファイル math.h をインクルードしている．さらに，これをコンパイルするためには，

```
gcc -o keisan sankaku.c -lm
```

と lm とオプションを追加する必要がある．

リスト 3: 三角関数表を出力するプログラム

```
1  #include <stdio.h>
2  #include <math.h>
3
4  #define NSTEPS 361
5
6  void cal_function(double x[], double s[], double c[], double t[]);
7
8  /* ================================================================== */
9  /*              メイン関数                                            */
10 /* ================================================================== */
11 int main(void){
12   double x[NSTEPS], s[NSTEPS], c[NSTEPS], t[NSTEPS];
13   int i;
14
15   cal_function(x, s, c, t);
16
17   for(i=0; i<NSTEPS; i++){
```

```
18      printf("%f\t%f\t%f\t%f\n",x[i], s[i], c[i], t[i]);
19    }
20
21    return 0;
22  }
23
24  /* ================================================================== */
25  /*              関数計算                                              */
26  /* ================================================================== */
27  void cal_function(double x[], double s[], double c[], double t[]){
28    int i;
29
30    for(i=0; i<NSTEPS; i++){
31      x[i]=(double)i/NSTEPS*2.0*M_PI;
32      s[i]=sin(x[i]);
33      c[i]=cos(x[i]);
34      t[i]=tan(x[i]);
35    }
36
37  }
```

3 コンソール入出力関数 (16 章)

コンピューターのもっとも基本的な入出力装置であるキーボードとディスプレイをコンソール (console: 操作卓) と呼ぶ．これらのコンソール入出力を利用した関数のプログラムの学習をする．とは言え，諸君はこれらの関数はかなり使ってきている．ここではそれら内容を整理して，これらの使い方の復習を行う．

3.1 標準入力と標準出力

コンソール入出力のことを，標準入出力と言うことがある．この標準入出力には，ファイルの入出力先に指定ができ[2]，そのために名前が付いている．標準入力を `stdin`，標準出力を `stdout` と言う．これらの関係を，以下に示す．

キーボード → 標準入力 → `stdin`
ディスプレイ → 標準出力 → `stdout`

3.2 書式付き出力 `printf`(p.320)

3.2.1 ディスプレイへの出力

`printf()` 関数を使えば，簡単にディスプレイ (標準出力) に表示できる．括弧内の最初のクォーテーションで囲まれた部分が出力される．`Hello world` のプログラムでおなじみであろう．変数に格納されたデータ—整数や実数，文字，文字列—を表示させたければ，変換仕様 (p.320〜) を使う．教科書には，いろいろ書かれており，諸君にはわかりにくいだろう．実際には，

```
printf("%d+%d=%d\n", a, b, result);
```

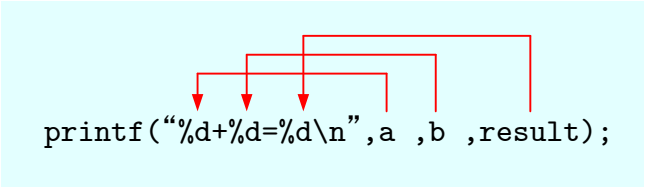

図 1: ディスプレイに表示させる printf 関数の意味

にように書く．この文の動作は，図 1 のとおりである．

ここで難しいのは，変換仕様の使い方である．本講義で使う変換仕様はそんなに多くなく，表 1 にまとめることができる．この中でも，文字列は滅多に使わないだろう．

あと，重要なことはエスケープシーケンス (p.27) である．エスケープシーケンスもいろいろあるが，'\n' と '\t' の使い方が分かれば，本講義では十分である．

表 1: 型に依存する変数定義や入出力

型	表示方法	変換仕様	備考
整数		%d	10 進数に変換
実数	浮動小数点	%f	小数点の表示．非常に大きな数値や小さい数値の表示には向かない．
		%20.15f	合計 20 カラムで，小数点以下 15 桁で表示
	指数表示	%e	非常に大きな数値や小さい数値を含む場合に都合が良い．
文字		%c	ひとつの文字を表示する場合に使う．
文字列		%s	文字列を表示させる場合に使う．

3.2.2 練習問題

書式付き出力 (printf) を使うと，任意の形でデータを出力できる．教科書の P.320〜325 を読み，以下の練習問題を実施せよ．

[練習 1] 円周率を，いろいろなフォーマットで出力せよ．ただし，円周率は math.h の中で M_PI で定義されている．それは，リスト 4 のようにすれば表示できる．

- 通常の%f で表示せよ．
- 小数点以下，3 桁で表示せよ．
- 小数点以下，10 桁で表示せよ．
- 指数形式で表示せよ．

[2] C 言語では，キーボードやディスプレイもファイルとして取り扱われる．

リスト 4 には，数学関数用のヘッダーファイル math.h が使われているので，コンパイルには-lm オプションが必要である．例えば

```
gcc -o fuga hoge.c -lm
```

とする．

[**練習 2**] 整数の 22446688 を 8 進数で表示せよ．また，16 進数で表示せよ．

[**練習 3**] 円周率と円周率の 2 乗の両方を小数点以下，10 桁で表示せよ．ただし，2 つの数値の間はタブ区切りとする．

リスト 4: π の表示

```
1 #include <stdio.h>
2 #include <math.h>
3
4 int main(void){
5
6   printf("%f\n", M_PI);
7
8   return 0;
9 }
```

3.3 書式付き入力 scanf(p.326)

3.3.1 キーボードからのデータの取り込み

キーボード入力の場合，書式付入力の scanf() という関数を使う方法が最も簡単である．この関数の引数は，ポインター[3]で，キーボードからのデータを入れる変数のアドレスを指定する．ポインターだのアドレスだのと面倒なことが多いが，そんなことが分からなくても，キーボードからデータの入力は可能なので安心してよい．

標準入力 (キーボード) からデータを読み込んで，それを変数 hoge に代入する場合，

```
scanf("%lf",&hoge);
```

と書けばよい．これは，

- scanf() は，キーボードからデータを入力するための関数．
- %lf は入力データが倍精度実数を表す．
- 変数 hoge はデータの格納先を表す変数である．変数の前に&を付けることを忘れてはならない．

というようなことを表している．図 2 のような感じである．

[3]値を代入する変数のアドレスのこと．

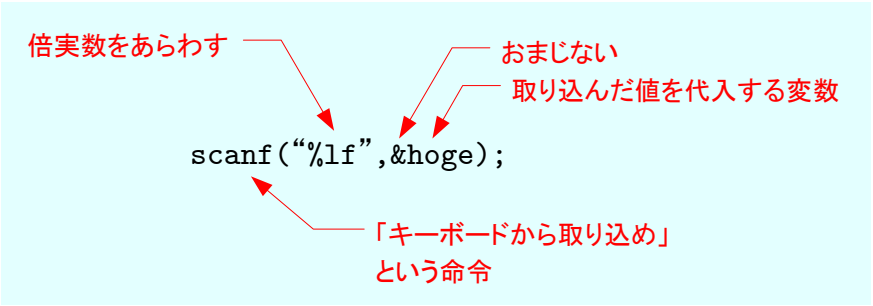

図 2: キーボードからデータを変数に取り込む scanf 関数の意味

キーボードからのデータの入力でも型を指定する必要がある．このようにコンピューターブログラムでは型というものが重要となる．これは，データをメモリーに格納する方法をプログラマーが指定する必要があるからである．これまで，使ってきた型指定の方法を表 2 にまとめる．ただし，倍精度実数で，非常に大きな数値や小さい数値の場合指数表示 (%e) を使う．

表を見て分かると降り，文字列は少し複雑である．幸いなことに，本講義では文字列を使うことはほとんどないので，この部分は余り気にしなくてよい．

表 2: 型に依存する変数定義や入出力

	整数	倍精度実数	文字	文字列
変数	`int hoge`	`double hoge`	`char hoge`	`char hoge[256]`
入力	`scanf("%d",&hoge)`	`scanf("%lf",&hoge)`	`scanf("%c",&hoge)`	`scanf("%s",hoge)`
出力	`printf("%d",hoge)`	`printf("%f",hoge)`	`printf("%c",hoge)`	`printf("%s",hoge)`
		`printf("%e",hoge)`		

3.3.2 改行文字の問題

データを入力する場合，データを入れた後に完了を示す [Enter] キーを押す．この [Enter] キーを示す文字'\n' も読み込まれようとして，悪さをすることがある．この，改行コード '\n' を読み捨てる必要がある．それを行うために，教科書 (p.335) には 3 通りの方法 (p.337) を示している．

- 数字を読み込んだあと，%*c で改行文字を読み捨てる．
- 数字を読み込んだあと，getchar() で 1 文字読み捨てる．
- 数字を読み込んだあと，gets() で残りの文字列を全て読み捨てる．

どれも一長一短がある．諸君が作成するプログラムは，諸君自身でしか使わないので，入力の処理にこだわらない方がよい．へんなデータを入れて，プログラムがクラッシュしても損害は無いからである．そこ

で，本講義では最も簡単な，最初の方法をとることにする．すなわち，キーボードから実数を読み込んで，変数 hoge に代入する場合，

```
scanf("%lf%*c",&hoge);
```

と書く．

3.3.3 プログラム例

リスト 5 にいろいろな型のデータを読み込んで，表示するプログラムを示す．このプログラムが理解できれば，コンソール入出力の基本は OK である．

リスト 5: いろいろな型のデータを読み込んで表示するプログラム

```
1  #include <stdio.h>
2
3  int main(void){
4    int i;
5    double d;
6    char c;
7    char s[256];
8
9    scanf("%d%*c",&i);
10   scanf("%lf%*c",&d);
11   scanf("%c%*c",&c);
12   scanf("%s%*c",s);
13
14   printf("\n\n");
15
16   printf("%d\n",i);
17   printf("%f\n",d);
18   printf("%c\n",c);
19   printf("%s\n",s);
20
21   return 0;
22
23 }
```

3.3.4 練習問題

[練習 4] キーボードから角度 [deg] を読み込んで，三角関数 (sin, cos, tan) の値を表示するプログラムを作成せよ．出力する三角関数の値は，1 行に表示しタブ区切りとすること．

3.4 お遊び

リスト 6 のプログラムの実行結果を予想せよ．分からない場合は，実行してみよ．

リスト 6: いろいろな型のデータを読み込んで表示するプログラム

```
1  #include <stdio.h>
2  #include <math.h>
3
```

```
4  #define NP 50
5
6  int main(void){
7    int i, iy;
8    double x;
9
10   for(i=0; i<=NP; i++){
11     x = 2*M_PI/NP*i;
12     iy = 30*sin(x)+40;
13     printf("%*c\n",iy,'*');
14   }
15
16   return 0;
17 }
```

4 課題

4.1 内容

以下の課題を実施し，レポートとして提出すること．

[問 1] (復) 教科書 [1] の第 12 章〜第 14 章 (pp.238–280) を 1 回読め．レポートには「1 回読んだ」と書け．

[問 2] (復) 教科書 [1] の第 15 章〜第 16 章 (pp.282–339) を 3 回読め．レポートには「3 回読んだ」と書け．

[問 3] (復) 本日配布したプリントを 2 回読め．レポートには「2 回読んだ」と書け．さらに，誤字・脱字，表現の悪いところ，間違いを指摘せよ．

[問 4] (復) 次のプログラムでは，何が表示されるか? また，`#define` の動作を述べよ．

```
#include <stdio.h>

#define NMAX 50

int main(void)
{
  int i, sum=0;


  for(i=1; i<=NMAX; i++){
    sum+=i;
  }

  printf("1+2+3+....+%d = %d\n", NMAX, sum);

  return 0;
}
```

[問 5] (復) 円周率を表すマクロ `M_PI` を使って，円周率を以下のようにさまざまなフォーマットで表示するプログラムを作成せよ．(ヒントプリント p.4 と教科書 pp.323–324)

```
3.14
3.141593
3.141592653590
3.14e+00
3.141593e+00
3.141592653590e+00
```

[問 6] (復) 次の関数の値を計算し，結果をディスプレイに表示するプログラムを作成せよ．

$$f(x) = \sin x \cos x + \sqrt{x^2 + \cos x}$$

表示する x の範囲は $[x_s, x_e]$ とし，N 分割して表示する．ようするに，ステップ $\Delta x = (x_e - x_s)/N$ で $f(x)$ の値を表示せよ—ということ．ただし，x_s と x_e，N はキーボードから読み込むこと．もちろん，x_s と x_e は倍精度実数，N は整数とすること．

[問 7] (復) リスト 6 のプログラムを実行せよ．そして，どうしてそのような結果になるか述べよ．

[問 8] (予) 教科書 [1] の第 17 章を 2 回読め．レポートには「2 回読んだ」と書け．

[問 9] ここでの学習内容でわからないところがあれば，具体的に記述せよ．

4.2 レポート 提出要領

期限	6 月 27 日 (水) AM 8:45
用紙	A4 のレポート用紙．左上をホッチキスで綴じて，提出のこと．
提出場所	山本研究室の入口のポスト
表紙	表紙を 1 枚つけて，以下の項目を分かりやすく記述すること． 授業科目名「計算機応用」 課題名「プリプロセッサーとコンソール入出力関数」 提出日 5E　学籍番号　氏名
内容	2 ページ以降に問いに対する答えを分かりやすく記述すること．

参考文献

[1] 林春比古. 新訂 C 言語入門 シニア編. ソフトバンク パブリッシング, 2004.