これまでの復習 (前期中間試験に向けて)

山本昌志*

2006 年 6 月 7 日

概 要

前期中間試験に向けて，これまで学習した内容をまとめる．このプリントは試験対策用である．

1 前期中間試験の傾向と対策

- 最初の講義の「情報モラル」は試験範囲外とする．
- 「情報モラル」を除いたプリントの内容が試験範囲となる．
- 教科書は，p.1–p.50 が範囲となる．1 章からの出題はほとんどないが，2〜3 回位は読み直した方がよい．
- このプリントの内容を良く理解すること．分からなければ，私を含めた他の人に聞くこと．

2 UNIX の基礎

2.1 ディレクトリー構造

UNIX を使う場合，以下のことを理解しておく必要がある．

- UNIX のハードディスク[1]のファイル構造は，図 1 のようにツリー (木) 構造呼ばれる階層構造になっている．それは，ファイルとディレクトリーから構成される．
- ハードディスクなどに記録されたデータのまとまりをファイルと言う．コンピュータが実行することができる命令の集合であるプログラムファイルと，コンピュータの利用者が作成した情報を記録しておくデータファイルがある．
- ファイルを分類・整理するための保管場所をディレクトリー[2]と言う．関連する複数のファイルをまとめて一つのディレクトリーに入れることにより，効率的に記憶装置を管理することができる．ディレクトリーの中にさらにディレクトリーを作成することもでき，階層構造によって細かい分類を表現することもできる．

*独立行政法人秋田工業高等専門学校電気工学科

[1]実際はハードディスクに限らず，CD-ROM やフロッピーディスクも，このツリー構造に含まれる．

[2]Mac や windows ではフォルダーと呼ぶ

- ディレクトリーには，以下のように表現されるものがある．
  - 今，自分が居るディレクトリーを，カレントディレクトリーと言う．[3]カレントディレクトリーを明示したい場合は，1 つのピリオド「.」で表す．
  - カレントディレクトリーの 1 つ上のディレクトリーを親デレクトリーと言う．2 つのピリオド「..」で，親ディレクトリーを表す．
  - カレントディレクトリーの 1 つ下のディレクトリーをサブデレクトリーと言う．複数存在することが可能なので，それぞれの名前で表す．
  - ユーザー各個人が使用 (読み，書き，実行) を許されている最上位のディレクトリーをホームディレクトリーと言う．
  - ログインしたときに入るディレクトリーをログインディレクトリーと言う．通常はホームディレクトリーと同じである．

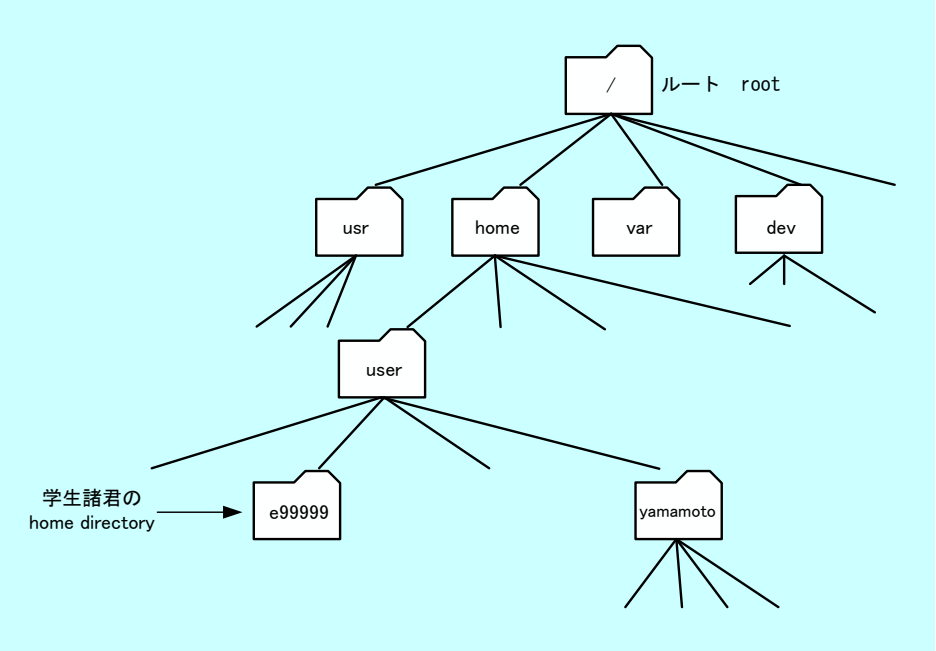


図 1: UNIX のファイル構造とそれを扱うコマンド

2.2 UNIX のコマンド

UNIX ではコンピューターに命令する場合，ターミナルからコマンドを打ち込む．ターミナルは，ユーザーがコンピューターに命令を行うウィンドウである．いろいろな命令があるが，まず諸君は以下の命令を覚える必要がある．

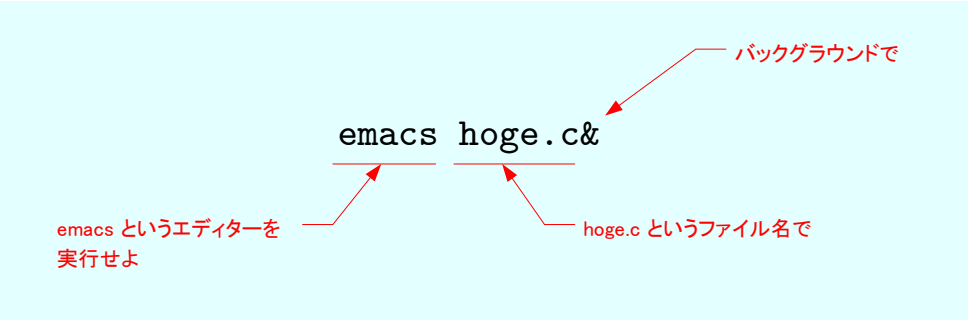
[3]ワーキングディレクトリーと言うこともある．

- カレントディレクトリー—自分がいる場所—を表示させるコマンドは,「pwd」である．このコマンドの由来は，print working directory である．
- カレントディレクトリーにあるファイルやディレクトリーを表示させるコマンドは,「ls」である．このコマンドの由来は，list である．
- ほかのディレクトリーに移動するときに使うコマンドは,「cd」である．このコマンドの由来は，change directory である．使い方は，次の通り．
  - 親ディレクトリーに移動する場合は,「cd ..」とする．
  - サブディレクトリー hogehoge に移動する場合は,「cd hogehoge」とする．
  - ホームディレクトリーに移動する場合は,「cd」とする．
- 人間のわかる C 言語のプログラムから，コンピューターがわかる機械語に変換する作業をコンパイルと言う．コンパイルを行うコマンドは,「gcc」である．hoge.c という C 言語のソースファイルから，fuga という機械語の実行ファイルを作るためには，以下のようにする．
  - 数学関数が含まれていない場合のコンパイルのためのコマンドは「gcc -o fuga hoge.c」である．
  - 三角関数や平方根などの数学関数が含まれている場合のコンパイルのためには，-lm というオプションが必要である．すると，コマンドは「gcc -lm -o fuga hoge.c」となる．

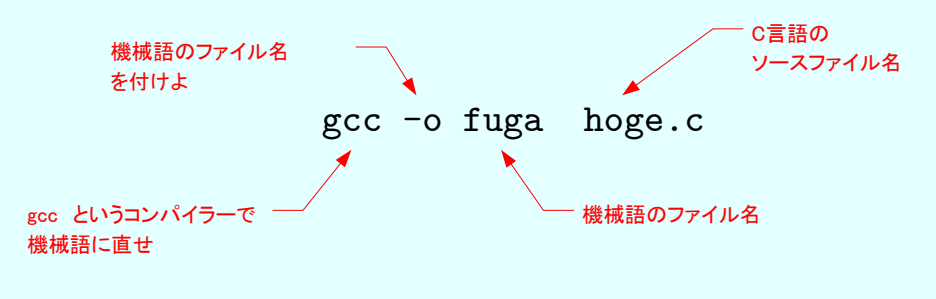
2.3 プログラムの作成順序

プログラムの作成順序は，図 2 のようになる．

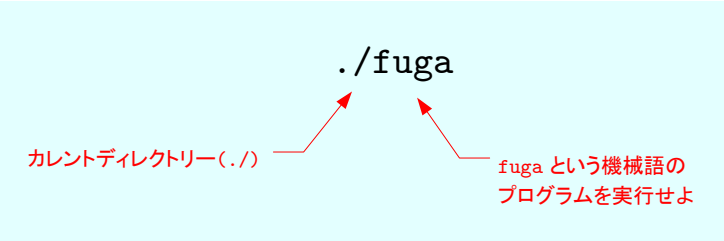
1. **作業用のディレクトリーの作成**　まずはプログラムを入れるのディレクトリーをつくる．プログラムをディレクトリー毎にわけることにより，大量のプログラムを管理する．デスクトップの「アカウント名+ホーム」と書かれたアイコンを開いて，右クリックの新しいフォルダーを選択することにより，ディレクトリーは作成できる．
2. **エディターによるソースプログラムの記述**　ターミナルをダブルクリックして開く．そして,「cd ディレクトリー名」をタイプすることにより，プログラム作成のディレクトリーに移動する．そして,「emacs ソースファイル名.c&」とタイプして，emacs を立ち上げプログラムを書きはじめる．最後の&は，emacs を動作させたターミナルから，コマンド入力ができるようにしている．



3. **コンパイル**　　プログラムを書き終えたら，それを保管する．そして，C 言語のファイルを機械語に変換する．変換するためには,「gcc -o 実行ファイル名　ソースファイル名.c」とタイプする．



4. **実行**　　コンパイルが完了したら,「./実行ファイル名」とタイプして，プログラムを実行する．



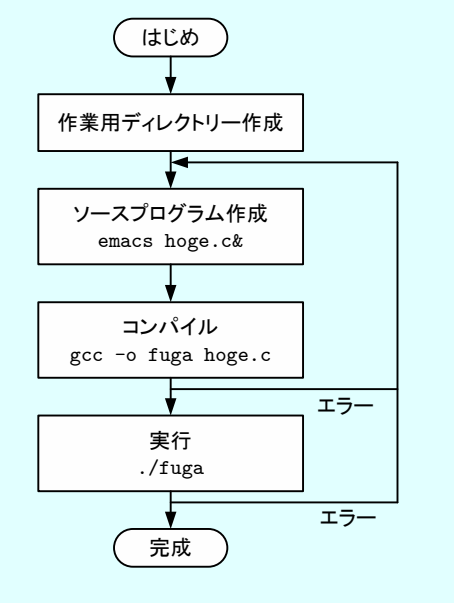
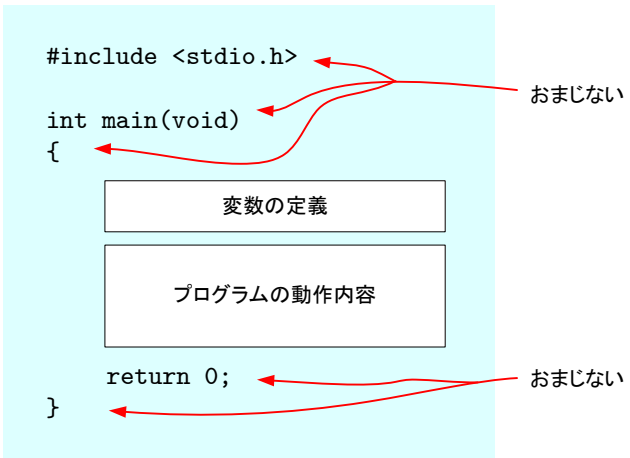


図 2: プログラムの作成手順

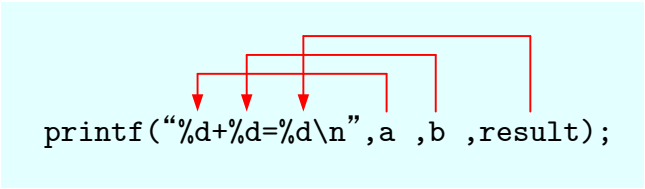
3 C 言語入門

- プログラムの書き方．プログラムは，下図に示すようにおまじないと変数の定義，動作内容を記述する部分から構成する．プログラムを作成するときは，おまじないの部分は気にしないでワンパターンで書く．プログラマーは動作内容を考える．



- printf() 関数の使い方．この関数は動作内容のひとつで，ディスプレイに文字やデータを表示させ

ることができる．



変数の値が a=10，b=3，result=13の場合

表示　10+3=13

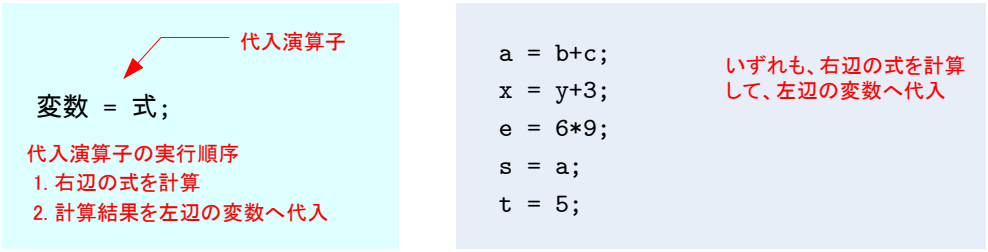
- \n は改行を，\t はタブを表すエスケープシーケンス (escape sequence) である．
- コメント文は，プログラムの動作にまったく関係が無い．プログラマーのためのメモである．/*〜*/で囲まれた部分はコメント文である．また，//を書くと行末までコメント文になる．
- データは変数の中に入れる．変数を使うためには，変数の定義が必要である．変数定義には型名と変数名を書く．整数型の型名は int，実数型の型名は double を使う．例えば，次のように変数の定義をすれば，整数を代入できる変数 i と j，hoge が使える．加えて，実数が代入できる変数 x と y，fuga も使える．

```
int i, j, hoge;
double x, y, fuga;
```

- 計算に使う代表的な演算子には，次のようなものがある．

C 言語の演算子	数学での意味
+	加算
-	減算
*	乗算
/	除算
%	余り

- 整数を整数で割る演算の結果，整数となる．小数部分は切り捨てられる．
- C 言語のイコール (=) は代入演算子と呼ばれ，数学のイコールとはまったく異なる働きをする．
  - 数学のイコールは左辺と右辺が等しい—ということを表している．
  - C 言語のイコールは右辺の式を計算して左辺の変数に代入する—という操作の命令を表している．したがって，左辺は変数で右辺は式となる．C 言語では数値がひとつの場合，あるいは変数がひとつの場合でも式と考える．値が評価できるものが式である．



- 整数を格納する変数—整数型の変数—hoge に，キーボードから整数を代入するためには，

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

  と書く．

  – 「scanf」は，キーボード[4]からデータと取り込め—という命令．
  – 「%d」は，キーボードのデータは 10 進数の整数[5]とみなす—ということを示している．
  – 「&hoge」は，キーボードからのデータは変数 hoge に格納する—ということを示している．

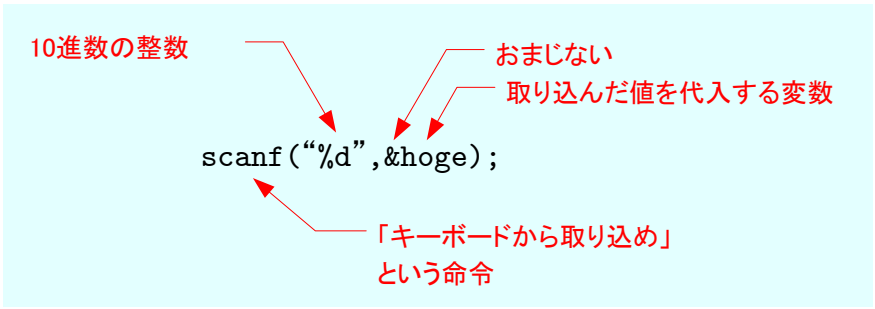


図 3: キーボードからデータを変数に取り込む scanf 関数の意味

- 実数を格納する変数—倍精度実数型の変数—fuga に，キーボードから実数を代入するためには，

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

  と書く．

- これまで，整数と実数の取り扱い方法を学んだ．コンピューターの世界では，実数と整数は明確に区別され，取り扱い方法が異なる．この辺のことは，今は分からなくてもよい．下表にしたがいデータが整数ときは整数を使う，実数のときは実数を使う—と憶えておく．

[4]教科書に書いてあるように，本当は標準入力．通常，キーボードが標準入力となっている．
[5]10 進数の整数を decimal number と言う．その頭文字の d が由来．

	整数	実数
変数の定義	int	double
キーボード入力　scanf	%d	%lf
ディスプレイ出力　printf	%d	%f

- 数学関数を使うときは，プログラムの先頭付近に`#include <math.h>`というおまじないを書く．
- 数学関数を使っているプログラムをコンパイルするときに，オプション`-lm`が必要である．すなわち，「`gcc -lm -o fuga hoge.c`」のようにする．
- 数学の $\sqrt{\ }$ の計算は，`sqrt()`と書く．
- `<math.h>`の中で，`M_PI`は円種率 (π) と定義されている．プログラム中で円周率を使う場合，`M_PI`と変数のように書けばよい．これが示す値は，3.1415 ...である．
- C 言語の三角関数は，全てラジアン単位である．π ラジアンが 180 度と憶えておく．そうすると，90 度は $\pi/2$ ラジアン，45 度は $\pi/4$ ラジアン，30 度は $\pi/6$ ラジアン，360 度は 2π ラジアンと，直ちにわかる．
- サインは`sin()`，コサインは`cos()`，タンジェントは`tan()`と書く．

4　課題の解答

4.1　プログラミング入門 (その 2)

[**問 1**]　略

[**問 2**]　略

[**問 3**]　次のようにすればよい．

```
 1 #include <stdio.h>
 2
 3 int main(void)
 4 {
 5
 6   printf("----------------------------------------\n");
 7   printf("Akita National College of Technology\n");
 8   printf("秋田工業高等専門学校\n");
 9   printf("\n");
10   printf("Yamamoto Masashi\n");
11   printf("山本昌志\n");
12   printf("----------------------------------------\n");
13
14   return 0;
15 }
```

4.2 プログラミング入門 (その 2)

[**問 1**] 略

[**問 2**] タブ (\t) を上手に使って，表を作成する．

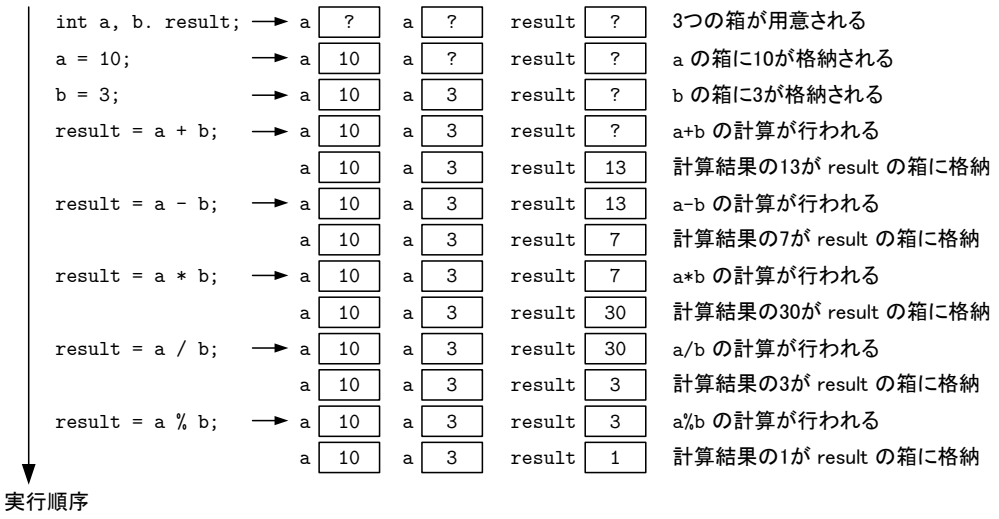
```
#include <stdio.h>

int main(void)
{

  printf("--------------------------------------\n");
  printf("補助\t読み方\t値\n");
  printf("======================================\n");
  printf("p\tピコ\t0.000000000001\n");
  printf("n\tナノ\t0.000000001\n");
  printf("m\tミリ\t0.001\n");
  printf("k\tキロ\t1000\n");
  printf("M\tメガ\t1000000\n");
  printf("G\tギガ\t1000000000\n");
  printf("--------------------------------------\n");

  return 0;
}
```

[**問 3**] 変数の変化は以下のようになる．



4.3 プログラミング入門 (おまけ)

[**問 1**] 略

[**問 2**] 略

[**問 3**] 四則演算には，+-*/を使う．余り (剰余) には%を使う．% を表示させるときには，%%と書く．

```c
#include <stdio.h>

int main(void)
{
  int a, b, c, d, e, f, g;

  a=543;
  b=123;

  c=a+b;
  d=a-b;
  e=a*b;
  f=a/b;
  g=a%b;

  printf("%d + %d = %d\n",a,b,c);
  printf("%d - %d = %d\n",a,b,d);
  printf("%d * %d = %d\n",a,b,e);
  printf("%d / %d = %d\n",a,b,f);
  printf("%d %% %d = %d\n",a,b,g);

  return 0;
}
```

[**問 4**] 整数同士の割り算は，切り捨てになる．結果は，以下のようになる．

```
543 + 123 = 666
543 - 123 = 420
543 * 123 = 66789
543 / 123 = 4
543 % 123 = 51
```

4.4 プログラミング入門 (その 3)

[**問 1**] 略

[**問 2**] お金の計算は，次のようにすればよい．

```c
#include <stdio.h>

int main(void)
{
  int one, five, ten, fifty, total;

  printf("1円玉の枚数?\t");
  scanf("%d",&one);

  printf("5円玉の枚数?\t");
  scanf("%d",&five);

  printf("10円玉の枚数?\t");
  scanf("%d",&ten);

  printf("50円玉の枚数?\t");
```

```
17    scanf("%d",&fifty);
18
19    total = one+5*five+10*ten+50*fifty;
20    printf("合計金額は, %d円です. \n",total);
21
22    return 0;
23  }
```

[問 3] 回路 (a) の合成抵抗 R_a は，$R = \frac{R_1 R_2}{R_1+R_2} + R_3$ と計算できる．一方，回路 (a) の場合は，$R_b = R_1 + \frac{R_2 R_3}{R_2+R_3}$ となる．

```
1   #include <stdio.h>
2
3   int main(void)
4   {
5     double R1, R2, R3, Ra, Rb;
6
7     printf("R1の抵抗?\t");
8     scanf("%lf",&R1);
9
10    printf("R2の抵抗?\t");
11    scanf("%lf",&R2);
12
13    printf("R3の抵抗?\t");
14    scanf("%lf",&R3);
15
16    Ra=R1*R2/(R1+R2)+R3;
17    Rb=R1+R2*R3/(R2+R3);
18
19    printf("回路(a)の合成抵抗は, %fオームです. \n",Ra);
20    printf("回路(b)の合成抵抗は, %fオームです. \n",Rb);
21
22    return 0;
23  }
```

4.5 プログラミング入門 (その 4)

[問 1] 円面積 s から円の半径 r は，$r = \sqrt{s/\pi}$ と計算できる．

```
1   #include <stdio.h>
2   #include <math.h>
3
4   int main(void)
5   {
6     double s, r;
7
8     printf("円の面積?\t");
9     scanf("%lf",&s);
10
11    r=sqrt(s/M_PI);
12    printf("半径は, %fです. \n",r);
13
14    return 0;
15  }
```

[問 2] 余弦定理を使うと辺 x は，$x = \sqrt{a^2 + b^2 - 2bc\cos\theta}$ と計算できる．a と b は他の 2 辺，θ は辺のなす角である．C 言語の三角関数では，ラジアンの単位を使う．

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

int main(void)
{
  double deg, rad, x;

  printf("角度?\t");
  scanf("%lf",&deg);

  rad=deg/180.0*M_PI;

  x=sqrt(0.57*0.57+0.3*0.3-2.0*0.57*0.3*cos(rad));
  printf("xは，%fです．\n",x);

  return 0;
}
```