これまでのまとめ

山本昌志*

2004 年 6 月 4 日

中間テストに向けて、今まで学習した範囲をまとめる。このプリントの内容を十分理解して、中間テストに臨むこと。

1 用語

教科書の第 1 章で、重要な記述を、以下に示す。これらは、確実に内容を理解し、憶える必要がある。

- コンピュータに何をどのようにさせるかを記述したものをプログラム (program) と呼ぶ。(p.13)
  - コンピューター機材を「ハード ウェアー」と呼ぶ
  - コンピューターに仕事をさせるためのプログラムを「ソフトウェアーと呼ぶ」
- コンピュータが理解できる言葉は、機械語 (マシン語) だけである。それは、「$\cdots 0010110100110 \cdots$」のように、0 と 1 の数字が並んだものである (p.16)。これを、人間が理解するのは大変です。
- そこで、人間がわかりやすい言語 (たとえば C 言語) でプログラムを書き、それを翻訳して機械語に直すことが考えられた。翻訳のように単純作業はコンピューターが得意である。
- コンピューターに与える指示を書いたものがプログラムである。指示を与える言葉をプログラミング言語と言い、いろいろな種類がる。しかし、どの言葉でも最終的には機械語に翻訳して、コンピューターは指示通りに仕事をする。
- コンピュータは単純な仕事しか理解できないので、目的達成までの方法や道筋を細かく表現して、指示を与える必要がある。しかも、正確に伝えなければ、コンピュータは正しく仕事をしない。(p.20)
- 問題解決までの道筋のことをアルゴ リズム (algorithm) と言う。(p.26)
- コンピュータに仕事をさせるには、人間が少しでも書きやすい (理解しやすい) ようなプログラミング言語を使ってプログラムを作り、それをコンピュータがわかるように翻訳し、コンピューターでそのプログラムを動かす手順をとる。(p.30)
- C 言語は、翻訳という手続きを必要とするプログラミング言語である。このような言語は、共通したプログラム構築の流れがる。(p.30-36)

*国立秋田工業高等専門学校　電気情報工学科

1. プログラムを記述 (コーディング)
   - 問題を解くための手順 (アルゴリズム) を、プログラム言語で決められた文法で記述する。この作業をコーディングと言う。
   - コーディングの作業により記述されたものをソースプログラム、あるいはソースコードと呼ぶ。
   - 通常、ソースプログラムは、文字を入力しテキストファイルで保存できるテキストエディタで書く。
2. プログラムを翻訳 (コンパイルとリンク)
   - ソースプログラムを機械語に翻訳する作業をコンパイルとリンクと言う。
   - コンパイルはコンパイラーと言うソフトウェアーにより、ソースプログラムをオブジェクトプログラムに変換する。
   - リンクはリンカーと言うソフトウェアーにより、オブジェクトプログラムとライブラリーから、機械語の実行プログラムを作ります。
   - 実行プログラムが記述されているファイルを実行ファイルと言う。
3. プログラムの動作確認
   - UNIX の場合、実行ファイル名をタイプすることにより、プログラムを実行させます。
4. プログラムの誤りの修正 (デバッグ)
   - ソースプログラムの翻訳時や機械語のプログラムの実行時の誤りを修正する作業をデバックと言う。

2 UNIX

2.1 ファイル構造

- UNIX のファイル構造は、木構造と呼ばれる階層構造になっている。
- データやプログラム、あるいはサブディレクトリーを入れるもの[1]をディレクトリーと言う。

2.2 UNIX のコマンド

- UNIX のファイル構造は、ツリー (木) 構造です。
  - 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います。カレントディレクトリーのパス (位置) を調べるコマンドは、「`pwd`」です。
  - パスを表す場合、デイレクトリーの区切りには「`/`」(スラッシュ) を使います。
  - 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「`.`」で表します。

[1] Mac や windows ではフォルダーと呼ぶ

- 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います。親ディレクトリーへ移動するコマンドは、「cd ..」です。2 つのピリオド「..」が、親ディレクトリーを表します。
- カレントディレクトリーの直ぐ下のディレクトリーを子ディレクトリー、あるいはサブディレクトリーと言います。例えば、子ディレクトリー hogehoge に移動するコマンドは、「cd hogehoge」です。
- 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います。移動するコマンドは、「cd」です。
- 2 通りのパスの表し方があります。例えば、私のホームディレクトリー (/home/tea/yamamoto) にサブディレクトリー (sub_1)、そのサブディレクトリー (sub_11) は、次のように表すことができます。相対パスはカレントディレクトリーからの相対位置を表し、ここではホームディレクトリーとする。

  絶対パス　/home/tea/yamamoto/sub_1/sub_11
  相対パス　sub_1/sub_11　あるいは　./sub_1/sub_11

- カレントディレクトリーにあるファイルやサブディレクトリーの名前を調べるコマンドは、「ls」です。
- サブディレクトリ hogehoge の追加と削除のコマンドは、以下の通り。

  追加　mkdir hogehoge
  削除　rmdir hogehoge

  ただし、hogehoge の中に、ファイルやサブディレクトリーがある場合は

  ```
  rm -rf hogehoge
  ```

  とする。-rf はオプションと呼ばれるもので、r は recursive(再帰) の略で、f は force(強制) の略である。
- ファイルを削除するコマンドは、「rm hogehoge」です。これで、hogehoge と言うファイルが削除されます。
- 空っぽのサブディレクトリー (hogehoge) を削除するコマンドは、「rmdir hogehoge」です。サブディレクトリーに中身が有る場合は、「rm -rf hogehoge」とします。サブディレクトリーに含まれるもの全て削除されます。
- ファイルやディレクトリーをコピーするコマンドは、「cp hogehoge hugahuga」です。これで、hogehoge というファイルあるいはディレクトリーの hugahuga という名のコピーが作成されます。
- ファイルやディレクトリーを移動するコマンドは、「mv hogehoge ../hugahuga」です。親ディレクトリー (..) にサブディレクトリー hugahuga が無い場合、hogehoge が親ディレクトリーに移動して、名前を hugahuga に変更されます。もし、親ディレクトリー (..) にサブディレクトリー hugahuga がある場合、hugahuga のサブディレクトリーとして、hogehoge と言う名前で移動します。
- 以前使用したコマンドを呼び出す機能をヒストリー機能と言います。「↑」や「↓」で使用できます。同じような長いコマンドを何回も打ち込む手間が省けます。

- 重要なコマンドをまとめると、以下のようになります。

`pwd`	現ディレクトリー (カレントディレクトリー) のパス (位置) の表示
`ls`	ファイルとディレクトリーの表示
`cd`	ワーキングディレクトリーの移動
`mkdir`	ディレクトリーの作成
`rmdir`	空のディレクトリーの削除
`cp`	ファイルやディレクトリーの複製
`mv`	名前変更や移動
`rm`	ファイルやディレクトリーの削除
`cat`	ファイルの表示や連結
`more`	ファイルの内容を一画面単位で出力
`man`	コマンドのオンラインマニュアル
↑又は↓	history。以前のコマンドの表示を行う。編集可能である。
[ctrl]+`c`	プロセスの強制終了
[Tab]	補完機能

2.3 コンパイル・リンクと実行

- C 言語のプログラムが書かれたソースファイルには、「hogehoge.c」のように拡張子「.c」が必要である。
- C 言語のソースファイル「hogehoge.c」をコンパイル・リンクして、実行ファイル「hugahuga」を作成するコマンドは、次の通りである。

```
cc -o hugahuga hogehoge.c
```

- ターミナルに「実行ファイル名 (例えば、hugahuga)」を打ち込んで [Enter] キーを押せば、プログラムは実行されます。

3 C 言語の文法

3.1 基本

3.1.1 約束事

- C 言語では、大文字と小文字は、区別されます。変数名 hogehoge と Hogehoge、hoGehoge は異なります。
- コンピューター内部では、\(バックスラッシュ) と ¥ (円マーク) の取り扱いは全く同じです。
- 空白、改行、タブは、関数の間に好きなだけ入れることができる。

- { と } は対応しており、{ と } で囲まれた部分は、一つの処理のまとまりを表す。
- 処理のひとつは、;(セミコロン) で区切られている。
- 足し算は+、引き算は-、掛け算は*、割り算は/で表すことができ、( ) を使って、数学と同じように計算式を書くことができる。
- コメント文は、プログラムの内容をわかりやすくするために記述するものです。これは、人間のためのもので、コンパイラーは無視します。/*〜*/で囲まれた部分が、コメント文です。行をまたいでも、それは有効です。
- プログラムは main() 関数から実行される。
- 基本的には、プログラムは上から下へと処理の動作が行われる。

3.1.2 おまじない

- 当面、プログラムの最初と最後にか書かれる、以下の部分はおまじないとして憶えておきます。もう少し進んだときに、その内容は教えます。

```
#include <stdio.h>

int main()
{

   文;

   return(0);
}
```

3.2 printf() 関数

- "(ダブルクォーテーション) で囲まれた部分に、ディスプレイ[2]に表示したい文字列や Y_n で与えられる変換仕様を記述する。変換仕様は対応する変数の出力方法を決めるものである。
- Y_n に対する変換仕様は、以下の通りである。

整数	%d
実数値	%f
より精度の高い実数値	%lf
1 文字	%c

今回のテスト範囲では、%d と%f が使えればよい。

[2]正確には標準出力

- 整数の表示桁数を調整するときは、変換仕様%d の%と d の間に、表示したい桁数を記述する。
- 少数部の桁を調整するときは、変換仕様%f の%と f の間に、「. 桁数」と記述する。
- 改行したい場合は、改行したい場所に「\n」を記述する。

3.3 変数

- 変数は、値を入れておく箱のようなものである。1 つの変数に 1 個の値を入れておく (記憶) ことができる。
- 変数を使う場合、プログラムの最初に変数の宣言を行う必要がある。
- いろいろな変数の型があるが、以下のものがよく使われる。

変数の型名	変数の取り得る値
int	整数
float	実数
double	精度の高い実数

  今回のテスト範囲では、int と double が使えればよい。
- 変数に値を入れるためには、代入演算子 (=) をつかう。

3.4 scanf() 関数

- キーボードから、値 (文字や数値) を読み込むために、scanf() 関数を使う。読み込んだ値は、変数に格納される。
- scanf() 関数の記述の仕方は、次の通りである。

```
scanf("変換仕様の並び ", &変数名, &変数名, ···, &変数名);
```

- 変換仕様とは、キーボードから入力されたものがどのような値か判断するために使う。主なものは以下の通りである。

整数	%d
実数値	%f
1 文字	%c

- 変数名は、先頭に&をつける必要がある。

4　プログラム例

次のプログラムの内容を理解し、書けるようになっておくこと。

4.1　メッセージの表示

```
#include <stdio.h>

int main()
{
  printf("Hello.\n\n");
  printf("How are you.\n");

  return(0);
}
```

4.2　携帯電話の料金

```
#include <stdio.h>

int main()
{
  int base, time, number;

  base = 3000;

  printf("通話時間を入力してください [分単位]\n");
  scanf("%d", &time);

  printf("メール送信数を入力してください\n");
  scanf("%d", &number);

  printf("----- 使用料金 ------\n");
  printf("%d 円\n", base+10*time+5*number);

  return(0);
}
```