これまでのまとめと結果の表示について

山本昌志*

2004 年 5 月 14 日

1　本日の学習内容

本日は、以下のことについて、学習します。

- これまでのまとめ I(コンピュータープログラムとは)
- これまでのまとめ II(コンピューターの取り扱い)
  - 秋田高専のシステム
  - UNIX のファイル構造
  - UNIX コマンド
  - プログラムの作成とコンパイル、実行方法
- 2 章 データを入力して、結果を表示してみよう。
  - 2.1 結果を表示するということ
  - 2.2 結果を表示するプログラムを書く
  - 2.3 プログラムの詳細

2　これまでのまとめ I(1 章 プログラムってなんだろう?)

2.1　コンピューターにとってプログラムってどんなもの

- コンピューター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。
  - 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」
- 人が頭の中で考えているだけでは、コンピューターは働いてくれない。コンピューターにきちんと指示を与える必要がある。あいまいな表現はだめである。

*国立秋田工業高等専門学校　電気情報工学科

  - 人は、あいまいな指示でも適当に処理 (仕事) をしてくれる。ただし結果は、人によって異なる。
  - コンピューターは、処理すべき内容をきちんと伝える必要がある。そのため、結果はいつでも、どれでも同じ。

- コンピューターは、自分で考えて仕事をすることはできない。人が、きちんと指示を与える必要がある。その指示をプログラムと言う。

- コンピューターは融通はきかないが、仕事は高速で正確、文句も言わず何度でも繰り返し同じ仕事を続けてくれる。

- コンピューターが唯一分かる言葉は、機械語 (マシン語) である。それは、「10010110001111···」のように 0 と 1 が並んだものである。これを、人間が理解するのは大変である。

- 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。

2.2 プログラムに必要なこと

- 一つのプログラミング言語をマスターすれば、短期間で 2 つめ、3 つめのプログラミング言語を覚えることができる。

- どのように伝えればコンピューターが目的の仕事をすることができるかを考えるのは、コンピューターでもプログラムでもなく、「人間の作業」である。

- 「コンピューターに目的をどのように伝えるか」という考え方 (プログラムの考え方) が、プログラミングに最も必要な知識である。

2.3 プログラムの考え方

- コンピューターは単純な仕事しかできない。そのため、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。

- プログラム作成では、次のことが大切である。

  - 問題の曖昧なところをはっきりさせる。
  - 問題を細かく分解して整理する。
  - 問題解決までの道筋をまとめる

- 問題解決のための道筋をアルゴリズム (algorithm) と言う。

2.4 プログラムの作り方

- コンピューターにさせたい仕事はさまざまである。さまざまな仕事をさせるために、さまざまなプログラミング言語がある。
- しかし、コンピューターが理解できる言葉は、数字の 1 と 0 を組み合わせた機械語 (マシン語) だけである。
- そのため、人間が理解できるプログラミング言語から機械語に翻訳する必要がある。
- プログラムの作成手順は、次のとおりである。
  - エディターを使って、プログラムを書く。プログラムを書くことをコーディング (coding) と言う。記述したものをソースプログラム (ソースコード)、そのファイルをソースファイルと言う。ファイル名は、「ファイル名.c」でなくてはならない。
  - コンパイラー (実際はリンカーも) を使ってプログラムを機械語に翻訳する。翻訳の作業をコンパイル (実際はリンクも) [1]という。翻訳されたプログラムを実行プログラム、そのファイルを実行ファイルと言う。
  - プログラムを実行して、動作確認を行う。
  - もしプログラムに誤りがあれば、ソースプログラムを修正して、再度コンパイルを行う。このプログラムの修正の作業をデバック (debug) と言う。
- 問題解決のための道筋をアルゴリズム (algorithm) と言う。

3 これまでのまとめ II(コンピューターの取り扱い)

3.1 秋田高専のシステム

この授業の C 言語の学習では、UNIX というオペレーティングシステム (operating system) を使う。情報処理センターの namahage と言う名前の UNIX マシーン 1 台を 44 人が共有して、c 言語を学習する。しかし、皆さんが直接触るパソコンは、Windos NT である。ネットワーク回線により、Windows NT から namahage を操作するのである。ネットワークで接続されているため、電気工学科棟 2F の私の研究室からも namahage を操作でき、便利である。設定さえすれば、アメリカからでも、南極からでも、宇宙からでも namahage を操作できる。

windows から UNIX を操作するプログラムは、PC X サーバー呼ばれ、いろいろありますが、秋田高専では「Exceed」と言うものを使っています。

- UNIX マシーンにユーザーがアクセスを開始することを「ログイン」という。Windows のデスクトップの「Exceed」のアイコンをダブルクリックすることにより、namahage への接続が開始される。アカウントとパスワードを入力すればログインできる。

[1]環境によっては、この作業をメイク (make) やビルド (build) と言う。

- UNIX マシーンへのアクセスを終了することを「ログアウト」という。UNIX のデスクトップパネルの「Exit」ボタンを押せば、ログアウト完了である。
- windos NT を触れているが、通信を行って、UNIX マシーンを操作していることを理解することが重要である。

3.2 UNIX のファイル構造

- 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[2]をディレクトリーと言う。
- ファイルの場所 (パス) の指定は、
  /home/tea/yamamoto/c/linear/GaussJordan/pivot.c
  のように行います。これで、pivot.c という C 言語のソースファイルの位置を指定している。記号「/」(スラッシュと発音) がディレクトリーの区切りを表す。ディレクトリーのパスの指定も同じで
  /home/tea/yamamoto/c/linear/GaussJordan
  のようにする。
- 現在ユーザーが作業するために居るディレクトリーを、「カレントディレクトリー」、あるいは「ワーキングディレクトリー」と言う。

3.3 UNIX コマンド

以下のコマンドを学習したので、覚えておく必要がある。

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	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　編集可能
[ctrl]+c	プロセスの強制終了
[Tab]	補完機能

[2]Mac や windows ではフォルダーと呼ぶ

ここで、ディレクトリーの削除について注意を与えておく。空のディレクトリーは、

`rmdir` ディレクトリー名

で削除できる。しかし、ディレクトリーの中にサブディレクトリーがあったり、ファイルがあるとこのコマンドでは削除できない。その場合は、

`rm -rf` ディレクトリー名

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。

3.4 プログラムの作成とコンパイル、実行方法

プログラムは、分かりやすく分類するためにサブディレクトリーに置く。サブでレクトリーの作成から、ソースファイルの作成、コンパイル、実行までの手順は以下の通りである。

1. ホームディレクトリー内にサブディレクトリー「sample1」を作成し、そのディレクトリーに移動する。
   - まずは、ホームディレクトリーに移動する。コマンド「`cd`」とターミナルで入力する。
   - サブディレクトリーを作成する。コマンド「`mkdir sample1`」と入力する。
   - 作成されたディレクトリーに移動する。コマンド「`cd sample1`」 と入力する。
2. 作成されたサブディレクトリーの中に、「sample1.c」というソースファイルを作成する。
   - デスクトップの「LPEX Editor」のアイコンをダブルクリックして、エディターを立ち上げる。
   - するとファイル名の入力を促すダイアログが出てくるので、「sample1/sample1.c」と入力する。
   - エディターの編集画面が出てくるので、プログラムを打ち込む。
   - プログラムを打ち込み終わると、その内容のセーブ (保管) する。
3. ソースプログラムをコンパイルする。
   - コマンド「`cc -o sample1 sample1.c`」と打ち込みます。cc がコンパイルをしなさいという命令である。次の-o は出力先を示すオプションで、その後に機械語に翻訳されたファイル名を書く。最後に、ソースファイル名を書く。したがって、ここでは、ソースファイル「sample1.c」を翻訳して機械語の「sample1」が出来上がる。
   - もし、エラーがあればソースプログラムを再編集して、保管、コンパイルを行う。
4. 出来上がった機械語の実行ファイルを実行させる。
   - ターミナルで、実行ファイルがあるか確認をする。コマンド「`ls`」を打ち込むと、実行ファイル「sample1*」が表示される。ファイル名の後ろにアスタリスクが付いているのは、実行ファイルの印である。
   - コマンド「`sample1`」と打ち込こむことにより作成したプログラムを実行できる。