1 プログラム作成までの流れ

基本的にはどのようにやっても、プログラムが完成すればよいのですが、まとめの意味もこめて紹介します。

1.1 作業ディレクトリの作成

まずターミナルを起動して、

```
$ pwd
```

とすると、大学の PC の場合は

```
/home/（ユーザ名）/.pclinux
```

となっているはずです。ここが各自のホームディレクトリです。個人のファイルはこのホームディレクトリ以下に置くことになります。ホームディレクトリは ~（チルダ）で表されます。たとえば、どこのディレクトリからでも

```
$ ls ~
```

とすると、ホームディレクトリ、つまり大学の PC の場合は /home/（ユーザ名）/.pclinux の中身が表示されます。

前にこの時間に来て作業をした人は、はじめに、他の講義のファイルと混じらないように、計算数学 1 で使うディレクトリ（~/keisan1）を作成しました。プログラムの課題をこなすには、いくつものファイルを作成するので、keisan1 ディレクトリの中にさらに日付で作業ディレクトリを作成するのがよいでしょう。

```
~/keisan1/0413
~/keisan1/0420
~/keisan1/0427
~/keisan1/0511
      ......
```

というようにディレクトリを作成します。

例として、今日（5 月 11 日）のディレクトリを作り、そこに移動します。

```
$ cd
```

cd の後に空白を一つ入れ、TAB を押すと候補が表示されます（ディレクトリの数が多いと、表示するかどうか聞いてくるかもしれません）。移動したいディレクトリ名を忘れてしまっても TAB を押すと候補が表示されるので、ls コマンドで調べる必要はありません。そして

```
$ cd k
```

として、ここで TAB を押すと keisan1 と補完されます（他に候補がなければの話ですが）。このように TAB を押すと、補完できるところまでは補完され、複数の候補があるときはその候補が表示されます。キーボードから入力する文字の数は減りますし、存在しないディレクトリに移動しようとしてエラーが出るということもなくなるので便利です。とにかく TAB を押してみると楽ができるかもしれません。

keisan1 ディレクトリに移動し、

```
$ mkdir 0511
$ cd 0511
```

で作成した 0511 ディレクトリに移動します。今日の作業はここでします。毎回、作業するためのディレクトリをこのように作成するとよいでしょう。

1.2 プログラムの作成とコンパイル

```
$ emacs&
```

として、Emacs を起動し、C 言語でソースコードを作成します。ここでは説明のために、作成したソースコードを sample.c という名前にします。ソースコードを実行できるファイルに変換することをコンパイルといいます。出来た sample.c をとりあえずコンパイルするには、

```
$ gcc sample.c
```

とします。コンパイルによって作成されたファイルは a.out になります（実際に gcc を実行した場合は、ls で a.out が作成されたかを確認してみてください）。

コンパイルによって作成されたファイルがすべて a.out という名前では不便です。そこで、別の名前（たとえば、sample という名前）をつけるにはオプション o を使います。具体的には、

```
$ gcc -o sample sample.c
```

とします。こうすると、a.out という名前ではなく、sample という名前のファイルがコンパイルによって作成されます。

最後に、コンパイルによって作成されたファイル sample を実行します。

```
$ ./sample
```

とします。当然、a.out という名前なら

```
$ ./a.out
```

とします。

これで、プログラムが求めていた通りの動作をすれば、終わりです。実際には、コンパイルがうまくいかなかったり、たとえコンパイルはできたとしてもプログラムがおかしな動作をする（途中でエラーが出て止まる、無限ループでいつまでたっても終わらない)、アルゴリズムの間違いがあったために求めているのとは違う結果が出力される、ということが起こります。

この場合は、Emacs でソースコードの間違いを探して直します。直し終わったら、必ず保存をしてください。そうしないと、ソースコードのファイルにその変更が反映されません。

最後に、作成したプログラムが無限ループで止まらなかったときに強制終了する方法を説明します。それはターミナルで「Ctrl+C」を押します（コントロールキーを押しながら C を押す)。

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

/* 次のような形式のファイルを読み込んで */
/* inputmat[][] に保存する */
/* 1 2 3 4 */
/* 5 6 7 8 */
/* 9 10 11 12 */

/* 行列の大きさは以下の ROW、COLUMN で指定する */
#define ROW 3
#define COLUMN 4


int main(int argc, char *argv[]){
  /* argc は入力した文字列の個数を受け取る */
  /* argv[] は入力した文字列を受け取る */
  FILE *fp;

  int input1, input2;
  double inputmat[ROW][COLUMN];

  /* 第一引数に指定されたファイル名を */
  /* 使ってファイルを開く */
  if((fp = fopen(argv[1], "r")) == NULL){
    printf("ファイルが見つかりません :%s\n", argv[1]);
    exit(1);
  }
  for(input1 = 0; input1 < ROW; input1++){
    for(input2 = 0; input2 < COLUMN; input2++){
      fscanf(fp, "%lf", &inputmat[input1][input2]);
    }
  }
  fclose(fp);

  /* 以下はプログラムの本体 */
  /* 課題に応じて変更する */
  printf("入力した行列は次の通りです\n");
  for(input1 = 0; input1 < ROW; input1++){
    for(input2 = 0; input2 < COLUMN; input2++){
      printf("%f ", inputmat[input1][input2]);
    }
    printf("\n");
  }
  /* 課題に応じて変更するのはここまで */

  return 0;
}
```