プログラミング演習 A 教材 (#2)

■ プログラムの実行

一つプログラムを作って実行してみましょう。

□ 画面に文字を表示する（だけの）プログラム

Emacs を起動して、下のプログラムを入力してください。ファイル名は任意の名前で結構ですが、例えば hello.c とでもしておいてください。C 言語では大文字と小文字（ A と a など）は区別されますので注意してください。

```
#include <stdio.h>

main(){

  printf("My name is Enokida Yuuichiro!");

}
```

入力ができれば、まず保存をしてください。正しく保存できたかどうか、ls コマンドで調べ、cat コマンドで中身を確認してください。
次に cc コマンドでコンパイルします（このコンパイルという作業の意味は後述）。

```
cc2004(86)% cc hello.c
cc2004(87)%
```

その後 ls すると、a.out というファイルが増えていることがわかるでしょう。これを ./a.out として実行して下さい。画面に名前が表示されていくはずです。（以下、作業例。下線が入っている部分は自分でタイプしたところ。）

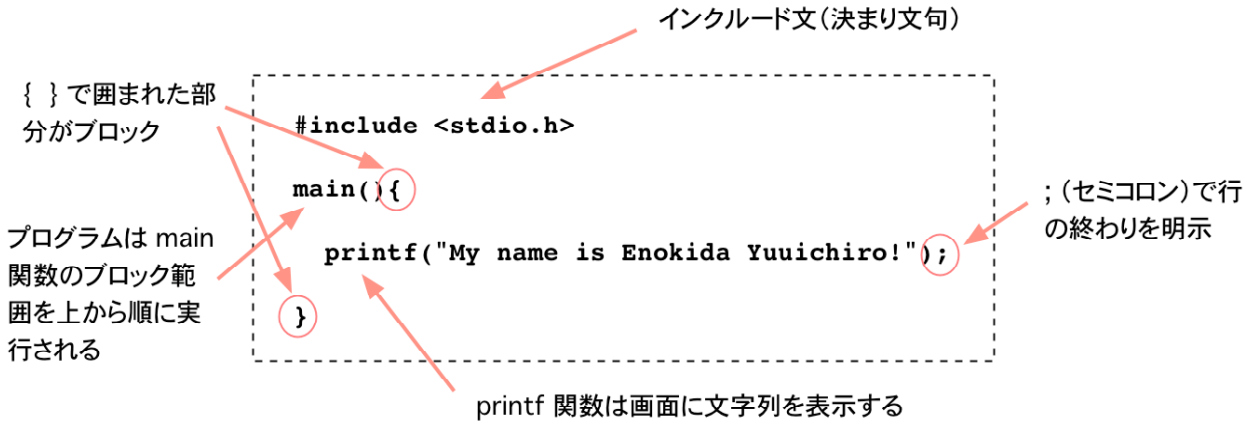
```
cc2004(86)% ls
Mail  Wnn6  public_html   hello.c   sample.txt
cc2004(87)% cat hello.c
#include <stdio.h>

main(){

  printf("My name is Enokida Yuuichiro!");

}
cc2004(89)% cc hello.c
cc2004(90)% ls
Mail  Wnn6  a.out     public_html   hello.c   sample.txt
cc2004(91)% ./a.out
My name is Enokida Yuuichiro! cc2004(92)%
```

それができたら、今度は Emacs で ”My name is …” の部分を自分の名前に変更して保存し直し、再び cc コマンドによってコンパイルし、./a.out によって実行して、変更が反映されていることを確認してください。

これが「C 言語でプログラムを作成し、実行する（そして修正してまた実行する）」という一連の作業の最も簡単な形になります。

□ プログラムの解説

プログラムの内容を行ごとに説明します。



- 1 行目の #include は今は決まり文句として覚えておいて下さい(*1)。
- { } によって囲まれた範囲をブロックと呼んでいます。
- 3 行目の main() 直後の { でブロックが始まり、最下行の } でブロックが終わっており、これが main() 関数(*2)の範囲を示しています。
- 5 行目の printf() 関数は画面に文字を表示する機能を提供します。
- 5 行目の最後には ；（セミコロン）があり、これが printf() 関数の行の終わりを示しています。
- プログラムは main() 関数の中身を上から順に一行ずつ実行します。

(*1) 「プログラミング C 」クラスで学びます。
(*2) 関数については何週か後で説明します。

□ わかりやすいプログラムの表記

2, 4,6 行目の空白や、5 行目の p rintf() 関数の前の空白は、なくても構いません。より見やすくなると思い、入れているものです。これらをすべて詰めてもプログラムとしては正しく機能します。つまり、上のプログラムはこのように書くこともできます。

```
#include <stdio.h> main(){printf(
"My name is Enokida Yuuichiro!");}
```

ただ、長いプログラムでこのような書き方をすると、プログラムが読みにくくなってしまいます。上に示したようにプログラムの中の文には構造があり、記号にも役割があるのですから、それらがよりはっきり分かりやすくなるよう、表記について努力すべきです。（別稿「コーディングスタイル」も参照のこと。）

□ 画面への出力を制御する

5 行目の printf() 関数と、その実行結果である画面上の表示に注目してください。

プログラム：

```
  printf("My name is Enokida Yuuichiro!");
```

実行結果：

```
My name is Enokida Yuuichiro!cc2004(92)%
```

プログラムによって出力された（画面上に表示された）”My name is …” の行のすぐ右にコマンドプロンプト（例では「cc2004(92)%」 ）が表示されて格好が良くありません。（通例、プロンプトは行の左端に表示されるものなので。）

これは printf() の出力指定に「表示したあとで改行する」という指示が含まれていないからです。改行表示がないと、出力は次々に右につながって表示されつづけます。

課題：
以下のような printf() の書き方をするとどのような出力結果になるか試せ
（各行の終わりを示す; を忘れないように注意。）

```
printf("My name ");
printf("is");
printf("Enokida Yuuichiro!");
```

C 言語での改行指定は、表示する文字列に「 \n 」（バックスラッシュと n ）を含めることで行います。具体的には以下のように書きます。

```
printf("My name is Enokida Yuuichiro!\n");
```

つまり「 \n 」は「改行するための文字(*3)」であり、これを画面上に出力すると、その次に出力された文字は次の行の左端から表示がはじまります。

(*3) 実際に C コンパイラは \n を \ と n という二文字としては扱わず、一文字（ASCII コードで 0a (10 番目)の文字）として扱っています。

□制御文字・エスケープシーケンス

こうした \ による特別な文字の表現は他にもあります。

\n -- 改行
\t -- タブコードを表示する
\\ -- \ を表示する
\' -- ' （シングルクォーテーション）を表示する
\" -- " （ダブルクォーテーション）を表示する

など。

課題：
以下のような実行結果になるよう printf() の記述を変更し、実行せよ

```
cc2004(97)% ./a.out
My name is
Enokida Yuuichiro!
I like "Sushi" much.
cc2004(98)%
```