13. 関数ポインタ

(1)プログラムのアドレス

コンピュータでは，電源投入直後，OS をデータとして読み込み，その先頭に制御を移す処理を行います。これが IPL(Initial Program Loader)です。

すなわち，システム記述用言語では，最低限，以下のことができなければなりません。

■プログラムが入っているアドレスを扱う。
■特定のアドレスに制御を渡す。

プログラムのアドレスも，C ではポインタとして扱います。

(2)関数ポインタ設定

これを宣言するには，関数のプロトタイプ宣言とデータのポインタ宣言のあいの子みたいな宣言を行います。

```
void (*key_mode)(char ch);
```

[意味]
① key_mode という変数に関数へのポインタが入る。
② 関数の引数は char 型。

関数へのポインタを設定するには，以下のように関数名を指定します。

```
void (*key_mode)(char ch);
void k_init(char ch);
   ⋮
main()
{   key_mode = k_init;
}
```

(3)ポインタによる関数呼出し

ポインタで指定されている関数を呼び出すには，次のように記述します。

```
key_mode = k_init;
   ⋮
(*key_mode)(ch);
 /* この表現は k_init(ch);と同等 */
```

ポインタによる関数呼出しを使わない場合，引数が同じで，異なる処理を行うには，たとえば以下のように記述するのが一般的でしょう。

```
switch(CD)
{
    case 0: fn00(ch);break;
    case 1: fn01(ch);break;
    case 2: fn02(ch);break;
    case 3: fn03(ch);break;
}
```

このような場合，関数へのポインタを使うことで，判定なしに関数を呼び出すことができます。

```
void (*fnX)(char ch);
fnX= fn02;
   ⋮
(*fnX)(ch);
```

(4)状態遷移図の制御

“/*”と“*/”で囲まれる部分を取り除く処理，すなわち，ソースプログラムから注釈を取り除く処理を考えてみましょう。なお，簡単化のため，文字列範囲を無視する処理は考えないことにします。