第1章 1.6 節で示したプログラムの入力関数 *Create_Input_Data* において必要となるであろう機能ブロック名の集合から 1 から *n* の正整数の集合への写像 *f* を定める方法を考える.

まず,

name	：	機能ブロックの名前を表す変数
Scanned_Names	：	入力された機能ブロック名の集合
counter	：	カウンタ(正整数) 機能ブロック名に 1 から *n* の番号を与えるために使う
f	：	求める写像

とし, 集合 *Scanned_Names* に対する辞書の操作を下記のようにする.

Member(*name, Scanned_Names*)	：	*name* が *Scanned_Names* に入っているか否かを true, false で示す
Insert(*name, Scanned_Names*)	：	*name* を *Scanned_Names* に入れる

さらに, 写像 *f* に対する下記の操作も利用する.

Assign(*f, name, counter*)	：	*f*(*name*) = *counter* とする
Number(*f, name*)	：	*f*(*name*) が既に定義されていたら, *f*(*name*) を返し, さもなくば 0 を返す

今, 集合 *Scanned_Names* および写像 *f* をそれぞれハッシュ表で覚え, 1 から *n* の各番号に対応する機能ブロック名(あるいは機能ブロック名へのポインタ)を覚える配列 *Block_Name*[・] も用意すれば, 指定された機能ブロック名 *name* に 1 から順に番号を付ける手順は, まず,

0-1˚ 集合 *Scanned_Names* および写像 *f* を共に空にする ;
0-1˚ *counter* := 0 ;

という操作を行った後, 機能ブロックの名前 *name* が入力されるたびに, 以下の操作を行えばよい. なお, 配列 *Block_Name*[・] は, 番号 *i* に対応する機能ブロック名を *O*(1) で見いだすことができるようにするためである.

```
{
1˚      if (  Member( name, Scanned_Names )  ) {
                /*  機能ブロック  name  には既に番号  i  が割り当てられている  */
1-1˚            i  :=  Number( f, name )  ;
1-2˚            機能ブロック  name  に対して必要な操作があれば, 実行する  ;
        }
2˚      else {
2-1˚            Insert( name, Scanned_Names )  ;
2-2˚            counter  :=  counter + 1  ;
2-3˚            Assign( f, name, counter )  ;
2-4˚            Block_Name[ counter ]  :=  name  ;
}       }
```

集合 *Scanned_Names* および写像 f をそれぞれハッシュ表で覚えれば,

Member(*name*, *Scanned_Names*)

Insert(*name*, *Scanned_Names*)

Assign(*f*, *name*, *counter*)

Number(*f*, *name*)

の実現は容易であろう.