第７回

２．構造体

構造体とは，同じ型に限定されない複数の関連するデータメンバの集合である。

・構造体の宣言　構造体指定子 struct を用いて

```
struct 構造体タグ名 {
    メンバ１の宣言；
    メンバ２の宣言；
    …
    メンバｎの宣言；
};
```

注）構造体タグ名は構造体の型名で，内容を定義するものでオブジェクトではなく，論理的なテンプレートである。

・構造体の変数の宣言　実際の記憶領域を占める物理的実体を確保する。

```
struct 構造体タグ名 変数リスト；
```

または構造体の宣言と合わせて

```
struct 構造体タグ名 {
    メンバ１の宣言；
    メンバ２の宣言；
    …
    メンバｎの宣言；
} 変数リスト;
```

あるいは

```
struct {
    メンバ１の宣言；
    メンバ２の宣言；
    …
    メンバｎの宣言；
} 変数リスト;
```

例. 構造体の宣言と初期化

```
struct sintai {
    char *name;
    int sinchou;
    int taijuu;
};
struct sintai a = {"Jack", 180, 66 },
              b = {"Betty", 172, 59 };
```

・構造体の参照　ドット演算子を用いて

構造体の変数名.メンバ名　　ex. a.taijuu

構造体配列の要素.メンバ名　　ex. constel[i].name

・構造体に対する演算　同じ型を持った構造体間では代入演算子が使用できてメンバ全部の複写ができる。

ex.　sa = sb;

・関数との関係　構造体は，ほかの型の値とまったく同じように，関数への仮引数として渡すことができる。また，関数の戻り値として，構造体を返すこともできる。

例. 構造体のメンバにアクセスするいくつかの方法を示す。

```
#include <stdio.h>

struct s_type {
    int i;
    char ch;
    double d;
    char str[80];
} s;

int main()
{
    printf("整数を入力して下さい：¥n");
    scanf("%d", &s.i);
    printf("文字を入力して下さい：¥n");
    scanf("¥n%c", &s.ch);
    printf("浮動小数点数を入力して下さい：¥n");
    scanf("%lf", &s.d);
    printf("文字列を入力して下さい：¥n");
    scanf("%s", s.str);

    printf("%d %c %f %s¥n", s.i, s.ch, s.d, s.str);

    return 0;
}
```

例. 構造体のサイズを調べる。

```
#include <stdio.h>

struct s_type {
    int i;
    char ch;
    int *p;
    double d;
} s;

int main()
{
    printf("s_type は %d バイトの長さである¥n", sizeof(struct s_type));

    return 0;
}
```

☆ s_type のサイズを求めるのに, sizeof(struct s_type)以外の書き方は?

・構造体へのポインタの宣言　構造体にアクセスするにはポインタを利用するのが一般的な方法である。構造体を指すポインタを用いて構造体のメンバにアクセスするときは，アロー演算子(->)を使用する。

例. 構造体を指すポインタの扱い方を示す。

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

struct s_type {
    int i;
    char str[80];
} s, *p;

int main()
{
    p = &s;

    s.i = 10;
    p->i = 15;
    strcpy(p->str, "I like structure");

    printf("%d %d %s¥n", s.i, p->i, p->str);

    return 0;
}
```

・ビットフィールド　構造体の便利な機能として，ビット単位でデータを取り扱うことができるビットフィールドがある。

構造体メンバとしてビットフィールドを定義するには次の書式を用いる。

型　名前：サイズ；

但し，「型」は int または unsigned

signed の場合，最上位ビットは符号ビットと見なされる。

移植可能にするには，signed あるいは unsigned と明示的に指定すべきである。

「サイズ」はフィールドでのビット数

ビットフィールドを参照するには，通常の構造体と同じように「.」または「->」演算子によってメンバを指定する。

例．ビットフィールドと他の型を混在させることが可能であり，またバイトやワードのすべてのビットが埋まるように定義する必要はない。

```
struct b_type {
    char name[40];              /* 品目名 */
    unsigned int instock: 1;       /* 在庫があれば 1，在庫切れなら 0 */
    unsigned int backordered: 1;  /* 未納注文であれば 1，納品済みなら 0 */
    unsigned int lead_time: 3;     /* 発注から納品までの月数（7 ヶ月まで） */
} inv[MAX_ITEM];
```

注）メモリのアドレス可能な最小単位はバイトであるため，ビットフィールド変数のアドレスを得ることはできない。

ビットフィールドには，ブール(真／偽)型のデータを格納することがよくある。

1 バイトに 8 個のブール型の値を格納できる。

《補足例題》a, b, c の３つのビットフィールドを持つ構造体(タグ名 b_type, 変数名 bvar)を使ったプログラムを作りなさい。a と b は 3 ビットの長さとし，c は 2 ビットの長さとする。また，すべて signed 変数とする。次に，bvar の a, b, c にそれぞれ値-1, 3, 1 を代入し，その値を表示しなさい。

```
#include <stdio.h>

main()
{
    struct b_type {
        signed int a: 3;    /* a, b, c それぞれ表現できる値の範囲は? */
        signed int b: 3;
        signed int c: 2;
    } bvar;

    bvar.a = -1;  /* -5 を代入するとどうなるか */
    bvar.b = 3;
    bvar.c = 1;   /* 3 を代入するとどうなるか */
    printf("%d %d %d¥n", bvar.a, bvar.b, bvar.c);
    printf("size of b_type = %d¥n", sizeof(struct b_type));
}
```

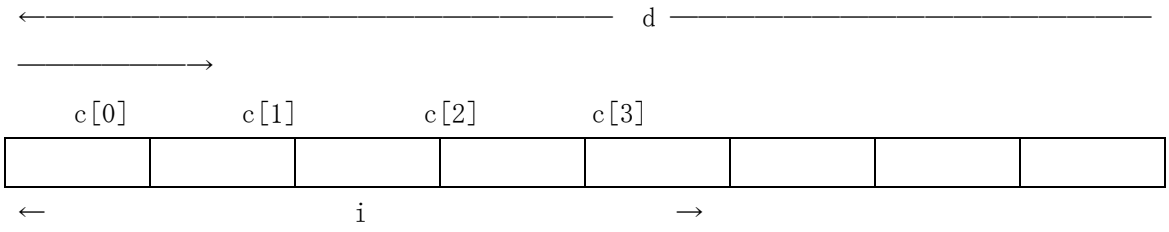
・**共用体**　複数のメンバが同じメモリの領域を共通して使用するように配置された単一のメモリ領域のことを指す。
メモリを共用する変数は，型が異なってもいいが，一度に１つの変数しか使用できない。共用体指定子 union を用いて次の書式で宣言する。

```
union 共用体タグ名 {
    メンバ１の宣言；
    メンバ２の宣言；
    …
    メンバｎの宣言；
} 変数名;
```

注）タグ名，変数名のいずれかを省略することができる。

例.

```
union u_type {
    int i;
    char c[4];
    double d;
} sample, *p;
```



共用体のメンバを参照するには，構造体と同様，ドット演算子もしくはアロー演算子を使用する。

《注意》

共用体のサイズは，最も大きいメンバのサイズとなるが，コンパイラがワード境界でそろえることがあり，sizeof 演算子で確認した方がよい。また，一度に１つの変数しか使用できないことに注意しよう。

```
#include <stdio.h>

main()
{
    union u_type {
        int i;
        char c[4];
        double d;
    } sample, *p;

    p = &sample;
    sample.i = 1234;
    printf("int i : %d¥n", sample.i);
    p->c[0] = '0';
    p->c[1] = 'K';
    p->c[2] = '¥0';
    printf("char c[4] : %s¥n", sample.c);
    p->d = 12.34;
    printf("double d : %f¥n", sample.d);
    printf("size of b_type = %d¥n", sizeof(union u_type));
}
```

例題３－２

生まれた月と日を入力し，星座が何であるかを出力するプログラムをつくりなさい。ただし，誕生日と星座の関係は次のとおりである。

山羊(12/23～1/20)，水瓶(～2/18)，魚(～3/20)，牡羊(～4/20)，牡牛(～5/21)，
双子(～6/21)，蟹(～7/23)，獅子(～8/23)，乙女(～9/23)，天秤(～10/23)，
さそり(～11/22)，射手(～12/22)

▼出力結果３－２

何月生まれですか？
3↵
何日生まれですか？
26↵

3 月 26 日生まれの
あなたは，牡羊座です

▼プログラム３－２

```
01      /* E3-2 */
02      /* 星座調べ */
03      #include <stdio.h>
04
05      struct {
06          char  name[9];
07          int   date;
08      } constel[] = { "山羊", 120, "水瓶", 218, "魚", 320,
09                      "牡羊", 420, "牡牛", 521, "双子", 621,
10                      "蟹", 723, "獅子", 823, "乙女", 923,
11                      "天秤", 1023, "さそり", 1122,
12                      "射手", 1222, "山羊", 1231 };
13
14      main()
15      {
16          int   i, month, day, nconstel;
17
18          printf("¥n何月生まれですか？¥n");
19          scanf("%d", &month);
20          printf("何日生まれですか？¥n");
21          scanf("%d", &day);
22          nconstel = sizeof constel / sizeof constel[0];  /* 配列の要素数 */
23          for (i = 0; i < nconstel; i++) {
24              if (month * 100 + day <= constel[i].date)
25                  break;
26          }
27          printf("¥n%d月%d日生まれの", month, day);
28          printf("あなたは, %s座です¥n", constel[i].name);
29      }
```

練習問題２５

２５−１．例題３－２のプログラムでは，12 月 32 日などが入力されると正しい結果が得られない。これを防止するため，誤った月日のデータが入力された場合は再度入力させるようにプログラムを変更しなさい。

《ヒント》各月が何日まであるかを格納した配列を用意して利用しなさい。

２５−２．前問１．において，構造体のメンバに対して，

```
constel[i].data
constel[i].name
```

と記述しているのを，->演算子によって記述するようにプログラムを変更しなさい。

《ヒント》構造体へのポインタを宣言して，これを用いなさい。

２５−３．姓，名，年齢，郵便番号，電話番号の各データをメンバとする構造体配列を下記のように宣言し，その構造体配列にキーボードからデータを逐次入力して，最大 100 名のデータを格納できるようにしなさい。但し，入力の終了は，姓入力の際に null と入力することによって判定しなさい。また，入力が終了した時点で，格納された複数名のデータをすべて出力するようにしなさい。

```
#define MAXREC  100

struct namelist {
    char   last_name[21], first_name[21];
    int    age;
    char   zip[9], tel[13];
} meibo[MAXREC];
```

▼出力例

```
Last name (終了は null 入力) > Natsume↵
First name > Soseki↵
Age > 49↵
Zip code > 123-4567↵
Telephone > 0120-1234567↵

Last name (終了は null 入力) > null↵

Name      : Soseki Natsume
Age       : 49
Zip code  : 123-4567
Telephone: 0120-1234567
```

２５−４．前問において，入力が終了した時点で，格納された複数名のデータを郵便番号順に出力するようにプログラムを変更しなさい。

《ヒント》構造体 struct namelist へのポインタおよび格納した人数を仮引数として，郵便番号順にデータの並べ替えを行う次のような関数

```
void namelist_sort(struct namelist *pnt, int count)
```

を作りなさい。

以前に解いた次の問題が参考になります。

==========

［参考］ 練習問題２０

ポインタの配列（*name[10]）を宣言し,次のような名前を 10 個セットしなさい。

″taro″, ″jiro″, ″hanako″, ″hirosi″, …… ″keiko″

この名前をアルファベット順に整列（並べ換え）させるプログラムをポインタへのポインタを使ってつくりなさい。

《ヒント》文字列の比較にはライブラリ関数の strcmp(s1, s2)を利用すればよい。

この関数の戻り値は,次のようになる。

s1 ＞ s2 のとき,　 0 より大きい整数

s1 = s2 のとき,　 0

s1 ＜ s2 のとき,　 0 より小さい整数

==========

２５−５．（難問）共用体を用いて，2 バイト整数の上下バイトを入れ換えて暗号化する関数 short encode(short)を作り，キーボードから入力した整数を暗号化／復号化するプログラムを作りなさい。但し，main 関数の部分は次のコードを用いなさい。

```
/* 関数のプロトタイプ宣言 */
short encode(short);

main()
{
    int n, m, k;

    while (1) {
        printf("整数を入力して下さい(終了は 0) ＝ ");
        scanf("%d", &n);
        if (n == 0) break;
        m = encode(n); /* 暗号化 */
        printf("¥n¥t%d を暗号化すると %d になります。¥n", n, m);
        k = encode(m); /* 復号化 */
        printf("¥n¥t%d を復号化すると %d になります。¥n¥n", m, k);
    }
}
```

▼出力例

整数を入力して下さい（終了は 0）＝ 123↵
123 を暗号化すると 31488 になります。
31488 を復号化すると 123 になります。

整数を入力して下さい（終了は 0）＝ -123↵
-123 を暗号化すると-31233 になります。
-31233 を復号化すると-123 になります。

整数を入力して下さい（終了は 0）＝ 0↵

２５−６．（超難問）整数係数の多項式について，その次数と係数をキーボードから入力して連結リストに格納してみましょう。次に，格納された多項式を画面に出力します。最後に，X の実数値をキーボードから入力して多項式の値を求めてみましょう。多項式の次数と係数を表現する構造体，および必要な関数のプロトタイプ宣言を次に示します。出力例を参考にして，プログラムを完成してみましょう。

```
typedef struct Node {
    int pow;   /* 次数 */
    int coe;   /* 整数係数  */
    struct Node *next;   /* 次のノードを指すポインタ */
} node;
node *start = NULL;   /* 先頭ノードを指すポインタで最初は空，外部変数 */

void insert(int pow, int coe);    /* 次数と係数をノードとして挿入 */
void show_poly();              /* 連結リストに格納された多項式の表示 */
double horner(double x);       /* 実数 x に対する多項式の値の算出 */
```

▼出力例

```
整数次数の多項式を入力します！
最大次数を入力してください：5↵

小さい次数から整数係数を順に入力してください

X^0 の整数係数：-1↵
X^1 の整数係数：2↵
X^2 の整数係数：3↵
X^3 の整数係数：0↵
X^4 の整数係数：0↵
X^5 の整数係数：-1↵
```

多項式は次の通り
-X^5+3X^2+2X-1

X の値を入力してください：0.8↵
多項式の値は 2.192320 です

《ヒント》次のプログラムは，キーボードから整数を順に読み込んで，それらを連結リストに格納して，最後にそれらを入力と逆順に表示するものです。これを参考に，連結リストの扱いを理解して，上記問題のプログラムを作成してみなさい。

```
/* キーボードから整数を読み，連結リストに格納 */
#include <stdio.h>
#include <stdlib.h>

typedef struct Node {
    int num;
    struct Node *next;
} node;
node *start = NULL;

void ins(int x);

main()
{
    int x;
    node *p;

    printf("整数を空白で区切って入力してください¥n");
    printf("終了には数字以外の文字を入力します¥n¥n");

    while (scanf("%d", &x) == 1)
        ins(x);
    printf("¥n 入力された数字を逆順に表示します¥n");
```

```
    for (p = start; p != NULL; p = p->next)
        printf("%5d¥n", p->num);
}


void ins(int x)
{
    node *p = start;
    start = (node *)malloc(sizeof(node));
    if (start == NULL)
        puts("メモリ不足です¥n"), exit(1);
    start->num = x;
    start->next = p;
}
```