C プログラミング 入門

— 構造体 —

早稲田大学

今回の目標

- 構造体を理解し作れる
  - 構造体とは何か
  - 構造体の作り方・使い方
  - メンバの初期化・参照

構造体と配列

【配列】

- 配列は 1 つ以上の変数を集められるデータ構造
- ただし，変数の型は同じものでなければならない

テスト点数

91	78	69	84	86	92	77	81	90

データ構造

int 型

【構造体】

- 構造体とは 1 つ以上の要素を集めて作られるデータ構造
- 配列のように全要素が同じ型である必要はなく，異なる型の要素を自由に組み合わせられる．

学生

氏名
得点
偏差値

学生 1	学生 2	学生 3	...
Aki	Yuu	Kou	
90	94	72	
64.2	67.1	56.3	

データ構造

char 型
int 型
double 型

構造体の使い方

- 構造体の型を宣言し，この型をもつ変数名を宣言することにより利用できる

構造体の型の宣言

【型】

```
struct 構造体タグ{
        型   メンバ名;
        ⋮
};
```

【使用例】

```
struct student{
        char   name[10];
        int    score;
        double   devi;
};
```

- 構造体タグは宣言した構造体の型を代表する識別子
- メンバが構造体を構成する変数

構造体変数の宣言

- 宣言した構造体の型をもつ変数をプログラム中で利用するために，新たな変数をその構造体の型として宣言する必要がある．

```
struct student{
        char   name[10];
        int   score;
        double   devi;
};

int main(void){
    struct student a, b;
    :
}
```

メンバの初期化

- 初期化は配列と同様．各メンバに対する初期値を，先頭から順に並べる．
- 与えられていない要素は，0 で初期化される．

```
struct student{
        char   name[10];
        int   score;
        double   devi;
};

int main(void){
    struct student a ={"Aki", 90, 64.2};
    :
}
```

メンバの参照

- 構造体の各メンバを参照するには，構造体メンバ演算子 (.) を利用する．

```
struct student{
        char name[10];
        int   score;
        double   devi;
};

int main(void){
    struct student a, b;
    strcpy(a.name,"Aki"); //文字列のコピー
    a.score      = 90;
    a.devi       = 64.2;
    :
}
```

構造体型変数の代入

- 代入演算子 (=) を用いて，構造体全体を一括してコピーできる．
- ただし，両辺の構造体の型は同一でなければならない．
- 下記プログラムを「struct1.c」とし実行せよ．

```
#include<stdio.h>
#include<string.h>
struct student{
        char name[10];
        int   score;
        double   devi;
};
int main(void){
    struct student a, b;
    strcpy(a.name,"Aki");
    a.score     = 90;
    a.devi      = 64.2;

    printf("Name: %s¥n", a.name);
    printf("Score: %3d¥n", a.score);
    printf("Devi: %.1f¥n", a.devi);
    :
    : 右へ続く
```

```
    :
    :
    b=a;     /*a を b へ代入*/
    printf("Name: %s¥n", b.name);
    printf("Score: %3d¥n", b.score);
    printf("Devi: %.1f¥n", b.devi);

    return 0;
}
```

【実行結果】

```
Name:  Aki
Score:  90
Devi:  64.2
Name:  Aki
Score:  90
Devi:  64.2
```

構造体を使ったプログラム

- 構造体を使ったサンプルプログラム
- 代入が可能な構造体は，関数の戻り値として使うこともできる
- 下記プログラムを「struct2.c」とし実行せよ（struct1.c をコピー）

```
#include<stdio.h>
#include<string.h>
/*構造体の宣言*/
struct student{
        char name[10];
        int   math;
        int   phys;
        double   ave;
};


/*平均値を計算する関数*/
struct student Average(struct student temp){

      temp.ave
     = (double) (temp.math + temp.phys) /2;
      return temp;

}

int main(void){
    struct student S1 = {"Aki", 90, 83};

    S1 = Average(S1);

    .
    . 右へ続く
```

```
    .
    .
    .

    printf("Name: %s¥n", S1.name);
    printf("Math: %3d¥n", S1.math);
    printf("Phys: %3d¥n", S1.phys);
    printf("Ave: %.1f¥n", S1.ave);

    return 0;

}
```

【実行結果】

```
Name: Aki
Math:  90
Phys:  83
Ave: 86.5
```

演習

課題 1：次のプログラムを作成せよ（TriArea.c）

- 直交座標系の 2 点 A_1, A_2 の x 座標と y 座標をキーボードから入力し，三角形 OA_1A_2 の面積を求めるプログラムを作成せよ．ただし点 O は原点 $(0, 0)$ とする
- 点の x 座標，y 座標を構造体のメンバとせよ
- すべて main 関数内で行うこと（関数は課題 2 で用いる）
- 面積を求めるには以下の公式を使うこと：
  2 点 $A_1(x_1, y_1), A_2(x_2, y_2)$ に対して，三角形の面積 S は

$$S = \frac{1}{2}|x_1y_2 - x_2y_1|.$$

- 表示は以下のようにする．

  Input x1 = 1 【Enter】
  y1 = 1.5【Enter】
  Input x2 = 3.5【Enter】
  y2 = 1.5【Enter】
  三角形の面積 S = 1.875

演習のヒント

- 絶対値の計算は fabs() を使う.
- math.h のインクルードと，コンパイル時の-lm 忘れないように
- 例えば以下のような構造体を宣言：

```
struct point{

    double x;
    double y;
};
```

演習

課題 2：TriArea.c を TriArea1.c とコピーし以下の改良を加えよ

- 三角形の面積を求める以下の形式の関数 Measure を付け加えよ
- 関数は点の情報を持つ構造体を 2 つ引数にとれ
- 関数は面積を戻り値とするようにせよ

まとめ

- 構造体を理解し作れる
  - 構造体とは何か
  - 構造体の作り方・使い方
  - メンバの初期化・参照