ソフトゼミＡ　第5回

構造体

今回は複数のデータであっても同時に、また異なる型の変数もまとめて取り扱うことができる「構造体」について学んでいきます。

✤構造体について

まずは構造体の基本的な形を見てみましょう。

```
/*構造体の型を宣言するとき*/
struct 構造体タグ名{
    データの型 メンバ名;
    データの型 メンバ名;
    ・・・;
};

/*宣言された構造体を使うとき*/
struct 構造体タグ名 変数名;
```

構造体の型自体の名前を構造体タグと呼び、構造体の構成要素をメンバと呼びます。構造体の型を宣言するときには「}」のあとに「;」を付けることを忘れないようにしてください。

main 文の前で構造体の型を宣言しておくことで、main 文の中でほかの型の変数を宣言するときと同じように使えます。

```
int main(void){
   struct 構造体タグ名 x;
   x.メンバ名 = 1;
         ・
         ・
         ・
   return 0;
}
```

では、プログラムの例を見てみましょう。

```
#include<stdio.h>
struct student{
    int age;
    double height;
};

int main(void){
    struct student a,b;
    a.age=16;
    a.height=139.0;
    b.age=19;
    b.height=171.75;
    printf(“Alice %d 才、身長:%f c m¥n”,a.age,a.height);
    printf(“Bob %d 才、身長:%f c m¥n”,b.age,b.height);
    return 0;
 }
```

この例では、構造体タグ:student、メンバ:age,height、構造体変数名:a,b、で２人の学生の年齢と身長のデータをまとめています。構造体を使うと型の異なる変数をまとめて宣言することができます。

宣言後にメンバにアクセスするときには、「a.age」のように「変数名.メンバ名」ドット(.)を用いて表します。

また、メンバや変数に配列を用いることもできます。この場合には、メンバや変数を指定するときに、通常の配列と同様に[]を用いて表します。

```
struct student2{
    int age;
    double height[2];
};

struct student2 c,d[2];
```

例えば上のように宣言した場合、以下の９つの変数が使えます。

“c.age”	“c.height[0]”	“c.height[1]”
“d[0].age”	“d[0].height[0]”	“d[0].height[1]”
“d[1].age”	“d[1].height[0]”	“d[1].height[1]”

✤ 練習問題

1. 学生３人の英語とドイツ語のテストの点数を入力し、その平均点を出力するプログラムを作成してください。

2. 学生５人の名前と身長を入力して、身長が１８０ｃｍ以上の学生の名前とその身長を出力するプログラムを作成してください。ただし該当する学生がいない場合は「該当する生徒はいません。」と出力してください。

3. 学生５人の出席番号と５０ｍ走のタイムを入力し、最もタイムのよい学生の出席番号とタイムを出力するプログラムを作成してください。