構造体

異なる型をまとめる

今までは同じ型を複数作成していた→配列

場合によっては
異なる型をまとめて使いたい→構造体

構造体とは複数の異なる型をまとめて作られた型である

構造体の作り方

構造体を作る前に
生徒の情報（学年、名前、身長）のデータを
今まで通り、変数で用意すると

```
int year;//学年
char name[64];//名前
double stature;//身長
```

となるが全て関連したデータであるにもかかわらず、
1つ1つが別の変数として宣言されているため、あまりわかりやすくない。→構造体はこれらの型を一つにまとめる

構造体の作り方

構造体のプログラム

```
struct student{
   int year;
   char name[64];
   double stature;
};
```

1. 構造体の型にはstructを使用する。
2. 構造体の型名（student）をつける。

   この型名を構造体タグ名と呼ぶことがある

構造体タグ名について

作成した構造体それ自体の名前であり
厳密には型名ではないので注意

構造体の型の変数の宣言

ここまでで構造体の型を作成したので、
次は構造体を実際に使うために構造体の型の
変数を宣言する必要がある。

宣言の方法

struct student data;

studentを構造体タグ
dataを構造体変数

ここまでのまとめ

プログラム

```
//main関数の前に構造体の型を作ること
struct student{
   int year;
   char name[64];
   double stature;
};
int main(void){
   struct student data;
   return 0;
}
```

構造体の使い方

作った構造体の型にある要素（yearやstature）を使いたいときは、次のようにする

構造体変数名.要素名
例）data.year = 1;

配列は同じ型の変数を数字で区別するが構造体は要素名で区別する

構造体の特徴

特徴
構造体変数自体を変数として取り扱える

構造体同士で変数を代入することが可能

一方で、演算や比較は不可能

構造体変数同士で代入を行う例

```
struct student{
   int year;
   char name[64];
   double stature;
};
```

```
int main(void){
   struct student data1,data2;
   data1.year = 1;
   strcpy(data1.name,”AAA”);
   data1.stature = 160.5;

   data2 = data1;
   return 0;
}
```

結果

data2はdata1のデータを得る

typedefについて

構造体を作るときには必ずstructが必要だが
構造体タグを新しいタグとして一度に宣言する
方法がある。→typedef

typedefを使うと新しい型を宣言できる

使い方
typedef 新しい型の形 新しい型名

typedefを使った例

```
struct student_tag{
   int year;
   char name[64];
   double stature;
};

typedef struct student_tag student;
```

これを使うことで宣言が楽になる

```
struct student_tag data;
                ↓
student data;
```

もっと簡略化

typedefを使って型を定義するのが面倒なので
構造体タグと構造体型を1度に宣言する

```
typedef struct student_tag{
   int year;
   char name[64];
   double stature;
}student;
```

（さらに構造体タグも省略できる）

構造体の特徴2

構造体変数は構造体型の引数を使うことが出来、1度に複数の情報を渡すことができる。

例
student型の構造体変数を引数として受け取る関数

void student_print(student data)

プログラムの例

```
void student_print(student data){
   printf("[学年]:%d¥n",data.year);
   printf("[名前]:%s¥n",data.name);
   printf("[身長]:%f¥n",data.stature);
   return;
}
```

構造体とポインタ変数

```
#include <stdio.h>
#include <string.h>
typedef struct {
   int year;
   char name[64];
   double stature;
} student;
```

```
int main(void) {
   student data;
   student *pdata;
   pdata = &data;//初期化
(*pdata).year = 10;
//常変数モードへの切り替え
strcpy((*pdata).name,"MARIO");
// 通常変数モードへの切り替え
return 0;
}
```

前のページのプログラム解説

構造体の各要素は、宣言の時の順番通りに並んでおり、&演算子で求められるアドレスは、構造体の始めの要素のアドレスである

構造体のポインタ変数の場合も、
*記号で通常変数モードに切り替えるが可能。
ただし、.の方が優先されるので、()をつける。

(*構造体ポインタ変数名).要素名

違う書き方もあり
構造体ポインタ変数名->要素名

ポインタ型の引数をもつ関数

参照 （プログラム）
http://9cguide.appspot.com/16-02.html

構造体の配列

構造体を配列にすることができる
student data[10];

使い方も今までと同じ
data[1].year = 2;

参考文献

苦しんで覚えるC言語
複数の型をまとめる
http://9cguide.appspot.com/16-01.html
http://9cguide.appspot.com/16-02.html
http://9cguide.appspot.com/16-03.html