初級C言語講座Vol.5

ソートアルゴリズム

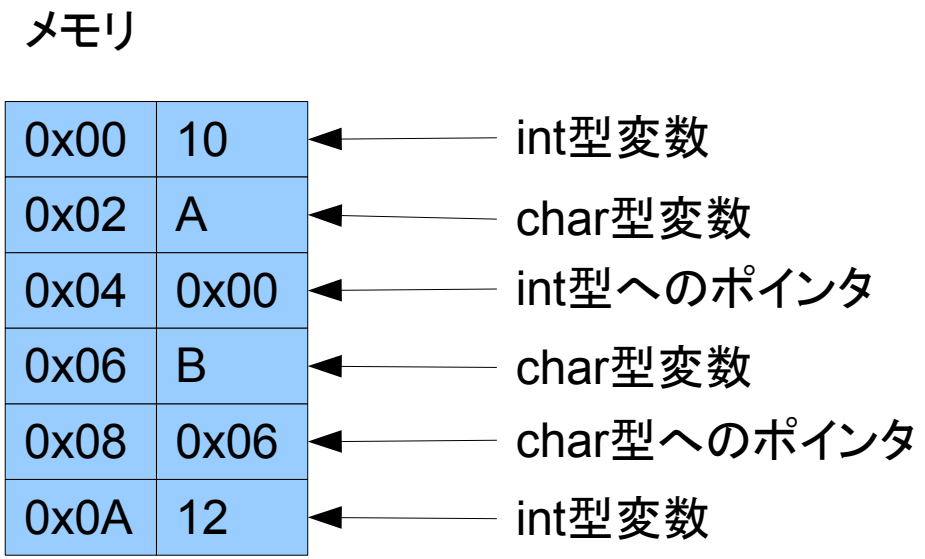
- ソートの手法にはいくつかある。
  - 選択ソート
  - バブルソート
    対象要素が、泡のように浮かんで整列していくことから名づけられたソート。でも速度が遅い。
  - クイックソート
    Cの標準関数にも用意されている、結構早めのソート
  - バケットソート
    条件にはまるとメチャ早。

ポインタ

- C言語を学ぶ上で、一番つまづく人が多い項目。
- でも、C言語でのコーディングで、これを利用しないというのはなかなか至難の業。

ポインタとは

- アドレスを用いて、間接的に変数などにアクセスする方法がある。
- ポインタとは、そのアドレスを保持する変数。



使い方

```c
#include <stdio.h>

int main()
{
    int a,b[10];
    int *p;   //ポインタは宣言時に*を付ける
    p=&a;     //アドレスを代入する。&とはアドレスを示す記号だったのだ！！
    printf("値を入力¥n=>");
    scanf("%d",&a);
    printf("入力された値は%d¥n",a);

    printf("値を入力(ポインタ)¥n=>");
    scanf("%d",p);                          //アドレスを渡すときはそのまま
    printf("入力された値は%d¥n",*p);        //その先にある値を使うときは*を使う。

    p=&b[0];                                //配列もこうやってアドレスを渡せる。
    p=b;                                    //先頭要素のアドレスなら、これでもよい。

    printf("値を入力(配列ポインタ)¥n=>");
    scanf("%d",(p+1));                          //&b[1]と同義
    printf("入力された値は%d¥n",*(p+1));//b[1]と同義

    return 0;
}
```

関数

- mainに山ほど書くと、見づらいプログラムになる。
- そこで、処理内容ごとにプログラムを分割する。
- 分割してまとめた部分を、関数という。
- ちなみに、main自体も関数である。
- 数学の関数と同じように、引数（n個）と
  返り値（1個）を持つ。

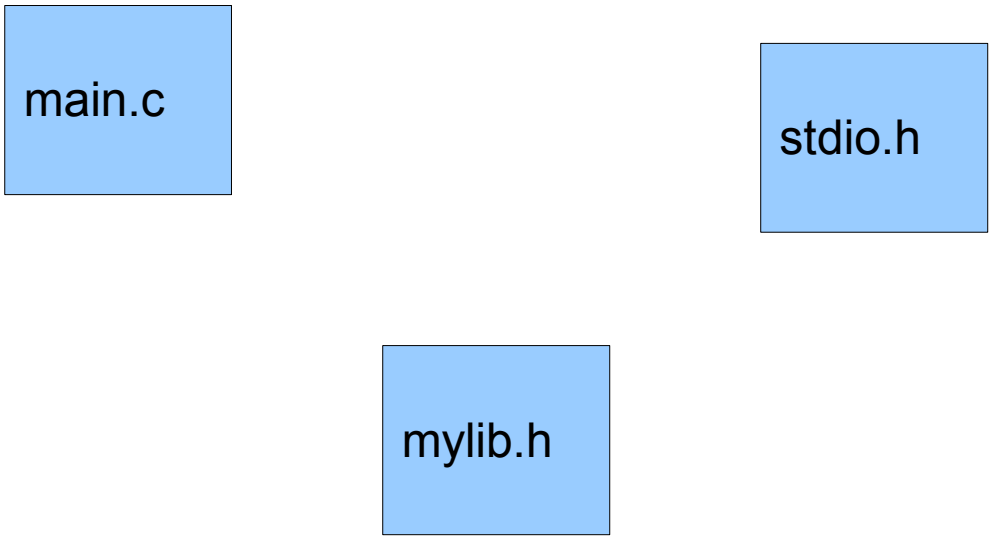
```
void swap(int *a,int *b)
{
     int temp;
     temp=*a;
     *a=*b;
     *b=temp;
}
```

使い方

```c
#include <stdio.h>

//関数の「プロトタイプ宣言」
int swap(int* , int*);

int main()
{
     int a=10,b=99,c;
     printf("a:%d b:%d",a,b);
     c=swap(&a,&b);
     printf("a:%d b:%d¥n成功？:%d",a,b,c);
     return 0;
}

//関数の中身
int swap(int *a,int *b)
{
     int temp;
     temp=*a;
     *a=*b;
     *b=temp;
     return 1;
}
```

複数ファイル

- ひとつのソースファイルに、大量に記述すると見づらい。
- 特定処理ごとに関数をまとめたり、あるいはよく使う関数をまとめてすぐに呼び出せるようにする。



使い方

mylib.h

```
//関数の「プロトタイプ宣言」
int swap(int* , int*);

//関数の中身
int swap(int *a,int *b)
{
     int temp;
     temp=*a;
     *a=*b;
     *b=temp;
     return 1;
}
```

main.c

```
#include <stdio.h>
#include "mylib.h"


int main()
{
     int a=10,b=99,c;
     printf("a:%d b:%d",a,b);
     c=swap(&a,&b);
     printf("a:%d b:%d¥n成功？:%d",a,b,c);
     return 0;
}
```

演習

- その1
  float型の値を二つ渡し、入れ替える関数を作れ。

- その2
  strcpy（文字列のコピー）と、
  同じ動作をする関数、mystrcpyを作れ。

演習

- その3
  配列にint型のデータを入力し、それを並べ替えて出力するプログラムを作れ。

  ただし、入力には通常の配列のアクセス方法、
  並べ替えと出力には、ポインタを用いたアクセスを行うこと。

課題　リスト構造

- 配列は、複数のデータを扱えるが、すでに整列している状態下では、中間に新しいデータを挿入することが難しい。
  そこで、リスト構造という概念を導入する。
- このようなリスト構造を作れ。
  また、次の関数も作れ。
  - データの挿入関数
    （指定した名前の後ろに挿入。無ければ、一番最後に）
  - 入れ替え関数
    （名前を二つ指定し、その要素を入れ替える。）

struct data[100]
struct data *next;
char str[100]

data[0]	data[1]	data[2]	data[3]	data[4]
&data[4]	&data[0]	&data[3]	&data[1]	&data[2]
CISC	ほへい	フェイス	あさげ	ねいむ