ソフトゼミ∀　第6回　関数

今回の∀ではAで収まりきらなかったstatic指定子と再帰呼び出しについて説明します。特に再帰呼び出しは今後使うことも多いと思いますので難しいとは思いますが頑張って習得しましょう。

補足（static指定子と再帰）

- static指定子
  関数内で定義した変数は関数が終了すると同時に破棄されますが、static 指定子を変数の定義の前に置くと関数が終了しても破棄されず値が保存されます。

a06_5.c

```
#include <stdio.h>
void add(void);
int main(void){
int i;
   for(i = 0;i < 5;i++){
      add();
    }
   return 0;
}
void add(void){
int a = 0;
   static int b = 0;              //最初の呼び出しの時のみ初期化
   a++;
   b++;
   printf("a = %d,b = %d¥n",a,b);
}
```

これをコンパイルして実行すると何となくつかめると思います。

- 再起呼び出し

  関数はその関数の中でその関数自体を呼び出すことができます。これを再起呼び出しといいます。いかに例を示します。

  a06_6.c

```
#include <stdio.h>
void print_number(int number,int n){
    if(number <= 0 && n <= 0){return;}
    else{
        print_number(number / 10,n - 1);
    }
    if(number > 0){
        printf("%d",number % 10);
    }
    else{
        printf("*");
    }
}
int main(void){
int number;
int n;
scanf("%d%d",&number,&n);
    if(number > 0 && n > 0){print_number(number,n);}
    else{printf("error.¥n");}
return 0;
}
```

  これはある数値と桁を入力してその数値の桁が入力した桁より小さいとき*で補うプログラムです。(正の値のみ)。print_number は number か n が 0 以上の時その値を 10 分の 1（小数点切り下げ）を渡してまた print_number を呼び出し、呼び出した関数が終わると受け取った値の下一桁を出力して終わります(number が 0 以下なら*を出力)。number、n の両方が 0 以下の時はそのまま終了します。以上の動作の繰り返しで出力します。

  再起呼び出しを使うと以上のようにソースをきれいに描くことができます。その反面メモリをよく消費する、無限ループに陥りやすいなどの欠点があるので注意が必要です。

追加問題

以下の関数を作成せよ。main 文を用意して動作確認をすること。

１、配列 a[n]の中の全ての値の合計を返す関数

int bsum(int a[],int n);

を作成せよ。

２、複素数を構造体で宣言し、2 つの複素数の値を乗算した複素数を返す関数

struct hukusum(struct a,struct b);

を作成せよ。

y=ｰ(ﾟ д ﾟ)・∵ﾀｰﾝＡ

y=ｰ(ﾟ д ﾟ)・∵ﾀｰﾝＡは練習問題、追加問題共に終わってしまい
余力のある方だけで構いません。もちろん出来なくても全く問題はありません。

１， n 番目のフィボナッチ数を返す関数

int fibo(int a);

を作成せよ。フィボナッチ数とは

f(0)=0
f(1)=1
f(n+1)=f(n-1)+f(n)

となる数である。(0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89…と続く)

（ヒント）再帰を使うと楽にできる。

また再帰を使うときは無限ループにならないようにどの条件で

再帰をやめるかよく考えること