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

普段変数は関数内で定義した場合、その関数が終了するとその変数は破棄されてしまいます。しかし static 指定子を使えば関数が終了してもその変数は破棄されず、値は保存されます。

a06_5.c

```
#include <stdio.h>
void add(void);
int main(void){
        inti;
        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);
}
```

上の例文でみると変数 add 内で宣言した変数 a は add が終了した時点で値は破棄されますが、変数 b は値は破棄されません。実行してみればそれとなく意味が分かると思います。

■ 再帰呼び出し

関数はその関数内でも関数を呼び出すことができます。これを再帰呼び出しといいます。

```
#include <stdio.h>
void print_number(intnumber,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。

再帰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]の中のすべての値の合計を返す関数
   intbsum(int a[],int n);
   を作成せよ。

2. 複素数を構造体で宣言し、二つの複素数の値を乗算した複素数を返す関数
   structhukusum(structa,sturuct b);
   を作成せよ。

## ◆ 応用問題 y=ー( ゜д゜)・∵. ﾀｰﾝA

**y=ー( ゜д゜)・∵. ﾀｰﾝAは練習問題、追加問題共に終わってしまい、余力のある人だけで構いません。**

1. ある村に大食いの怪物の子供がやってきた。
   大食いの怪物は村にあるものを食べて成長し、1日に一匹子供を産む。子供は一日で大人に成長し、また、大人になった次の日から一日ごとに子供を産み始める。
   n日後、村には何匹の怪物がいるか答えよ。

   ※再帰を使ってみよう！