ソフトゼミ∀第６回

# 問題の解答とおまけ

## ◆ ∀問題

以下の関数を作成し、main 文で動作を確認しなさい。それぞれ別々のファイルでよい。

1. double 型の配列 a、a の要素数 n、処理した配列の出力先の double 型配列 b を仮引数とする。配列 a についてその平均値より大きい要素は平均値に直して、平均値より小さい要素はそのままで配列 b に書き込む関数。ただし配列 a を書き換えてはならず、出力先の配列が必要な要素数を持つかは呼び出し側の責任とする。

※呼び出し元の配列は代入により用意してもよい。

```
#include<stdio.h>
void turna(double a[],int n, double b[]) {
	double ave=0;
	int i;
	for (i = 0; i < n; i++)ave += a[i];
	ave /= n;
	for (i = 0; i < n; i++) {
		if (a[i] >= ave)b[i] = ave;
		else b[i] = a[i];
	}
}
int main(void) {
	double x[5] = {1.2, 6.5, 2.1, 4.0, 3.1};
	double y[10];
	int i;
	for (i = 0; i < 5; i++)printf("x[%d] = %3.3f, ", i, x[i]);printf("¥n");
	turna(x, 5, y);  //ここで関数を実行している
	for (i = 0; i < 5; i++)printf("y[%d] = %3.3f, ", i, y[i]);printf("¥n");
	for (i = 0; i < 5; i++)printf("x[%d] = %3.3f, ", i, x[i]);printf("¥n");
	return 0;
}
```

プログラムで用いられた関数は void 型ですが、配列引数の特性を用いて出力先の配列を仮引数(ここでは配列 b)として用意し、呼び出し元で実際の出力先を用意して引数に指定(ここでは配列 y)することで疑似的に返り値とできます。他の型の関数にも応用可能です。

2. 4 つの int 型の変数 w、x、y、z を引数とし、大きい順に並び替えて printf 文で出力する関数。
※ヒント：「バブルソート」のアルゴリズムについてネットで検索するとよい

```
#include<stdio.h>
void sort(int w, int x, int y, int z) {
        int i, j, m[4],n=0;
        m[0] = w;
        m[1] = x;
        m[2] = y;
        m[3] = z;
        for (i = 0; i < 4; i++) {
                for (j = 3; j > i; j--) {
                        if (m[j] > m[j - 1]) {
                                n = m[j];
                                m[j] = m[j - 1];
                                m[j - 1] = n;
                        }
                }
        }
        printf("ソート結果：%d %d %d %d¥n", m[0], m[1], m[2], m[3]);
}
int main(void) {
        int a, b, c, d;
        printf("整数4つを入力：");
        scanf("%d%d%d%d", &a, &b, &c, &d);
        sort(a, b, c, d);
        return 0;
}
```

このプログラムでは、配列のうしろから順にひとつ前の要素と比較し、後ろの要素の方が大きい場合に入れ替える、という動作を繰り返し、一巡すると先頭が一番大きくなります。一番大きい先頭の要素を除き、比較するべき要素がなくなるまでそれを繰り返します。

## ◆ おまけ：再帰呼び出し

ゲーム作成にはほとんど使いませんが、おまけという枠で紹介させていただきます。ある関数の定義の中で、その関数自体を呼び出すことを再起呼び出しといいます。再起呼び出しを使うと、今回の練習問題の 3(階乗を求める関数)は以下のように書き直せます。

```
int kaijyo(int a) {
        if (a == 0)return 1;
        return a*kaijyo(a-1);
}
```

個の関数の戻り値は、引数が 0 なら 1、それ以外なら a×(a-1)!です。つまり、

a×(a-1)!

a×a-1×(a-2)!

…

a×a-1×a-2×…×1

を求めていることになります。最後呼び出された関数から順に戻り値を返していくので、戻り値は 1、2、6、24…の順で返されていき、最初の呼び出し元である別の関数にはしっかりと a の階乗が返されます。

アルゴリズム的な要素を多大に含むときに再起呼び出しは便利ですが、無限ループや思わぬ失敗を引き起こしやすいので注意が必要です。