プログラミング演習I第4回レポート

学籍番号：2364902

名前：キム　ギュソク

(1) 課題番号と課題内容

A. 基本課題1

i. 3 次元ベクトルの構造体を用い、この構造体を引数として 外積を返す関数を作成する。

(2) フローチャートまたは疑似言語によるアルゴリズムの記述

A. フローチャートは次のページの図1に記述している。

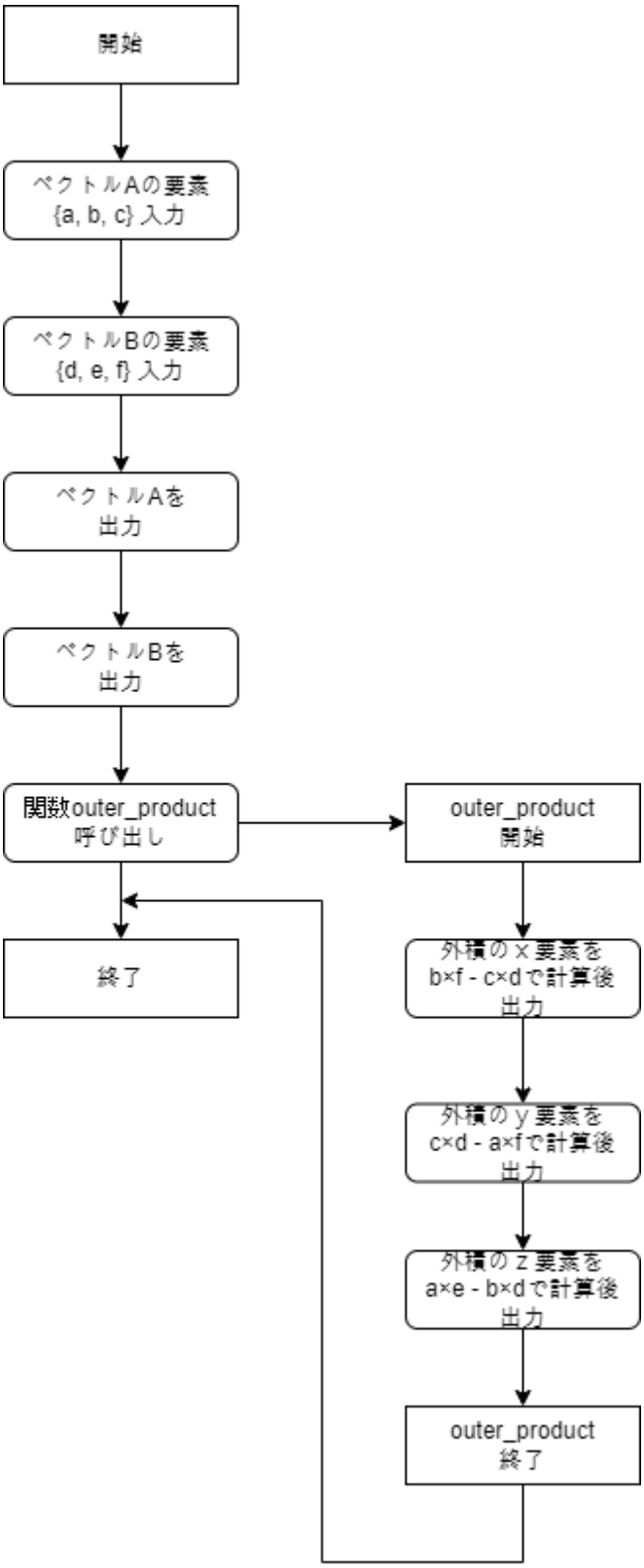

開始
ベクトルAの要素
{a, b, c} 入力
ベクトルBの要素
{d, e, f} 入力
ベクトルAを
出力
ベクトルBを
出力
関数outer_product
呼び出し
終了
outer_product
開始
外積のｘ要素を
b×f - c×dで計算後
出力
外積のｙ要素を
c×d - a×fで計算後
出力
外積のｚ要素を
a×e - b×dで計算後
出力
outer_product
終了

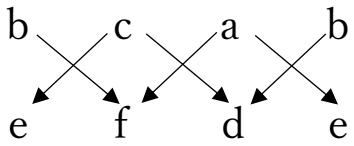
図 1。フローチャート

(3) アルゴリズムが正しいことの説明

A. 課題 1

i. 正当性

1. 理論的な外積の計算は　$\vec{A} = \{a, b, c\}$, $\vec{B} = \{d, e, f\}$ の場合次のような方法で求めることができる。



$$A \times B = \{bf - ce,\ cd - af,\ ae - bd\}$$

2. フローチャートでも書いているように外積の x の計算はbf − ceとして y の計算は cd − afとして z の計算はae − bdとして行ったのでこの計算結果は正当だと考えることができる。

ii. 停止性

1. 今回のプログラムでは反復処理を使ってないので無限ループに入ることはなくすべてのコードを実行すると停止することがわかるので。停止性も満足していると考えられる。

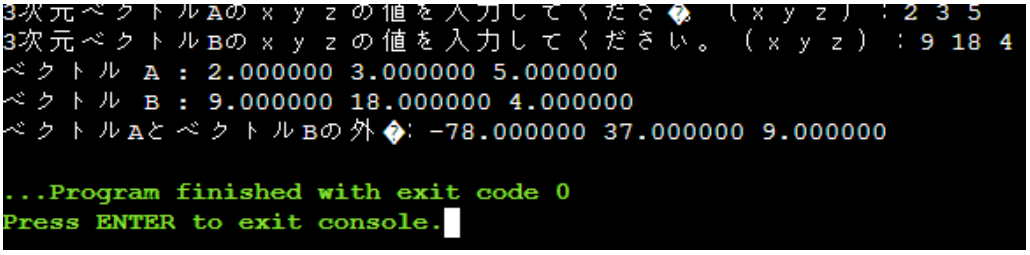
(4) ソース・プログラムの説明

ソースコード

main.c

```c
#include <stdio.h>

struct vector{//3dimensions vector
    double x;
    double y;
    double z;
};

void outer_product(struct vector arr1[], struct vector arr2[]);//outer_product function

int main(void){
    struct vector v1;
    struct vector v2;
    printf("3次元ベクトルAのｘｙｚの値を入力してください。（ｘｙｚ）：");//input vector A value
    scanf("%lf %lf %lf", &v1.x, &v1.y, &v1.z);
    printf("3次元ベクトルBのｘｙｚの値を入力してください。（ｘｙｚ）：");//input vector B value
    scanf("%lf %lf %lf", &v2.x, &v2.y, &v2.z);
    printf("ベクトル A : %lf %lf %lf\n", v1.x, v1.y, v1.z);//print vector A
    printf("ベクトル B : %lf %lf %lf\n", v2.x, v2.y, v2.z);
    printf("ベクトルAとベクトルBの外積：");
    outer_product(&v1, &v2);

    return 0;
}

void outer_product(struct vector arr1[], struct vector arr2[]){
    printf("%lf ", (arr1->y * arr2->z) - (arr1->z * arr2->y));
    printf("%lf ", (arr1->z * arr2->x) - (arr1->x * arr2->z));
    printf("%lf ", (arr1->x * arr2->y) - (arr1->y * arr2->x));
}
```

実行結果



A. ソースコードの説明

i. main関数を呼ぶ前に構造体の形式を宣言する。vectorという構造体を宣言して要素として座標が入るので座標の場合実数も入ることができるのでdouble型で宣言する必要がある。そして、main関数を呼び出して構造体v1とv2を宣言する。そして、Userから各ベクトルの要素を入力してもらう。その後確認として各ベクトルを出力する。そして、外積を計算して出力する予めプロトタイプで宣言した関数outer_productを呼び出す。outer_productの中では外積計算によって出てくるｘｙｚの要素を各自計算して出力する。

(5) 考察

A. 今回の外積の計算ではｘｙｚの要素の計算を一つ一つ計算をして出力をしたが、この計算は同じ作業の反復なのでfor文を利用して作ることが可能だと思う。しかし、この場合構造体の要素を参照することが連続的ではない為for文を利用して作ることは余計な条件が付くので非効率的である。よって、今回の外積計算は公式に従って直接該当する構造体の要素を呼び出す必要がある。また、ｘｙｚの要素を単純な配列に入れることもまた可能であるが、構造体を利用した理由としては配列を使った場合は何番目の要素の意味していることが分からなくなるのでコードの作成に困難が生じる可能性があるので入力した数字がどのような意味の変数に入ったかを確認できる構造体のほうがよりいいと思う。

(6) 感想

A. 普段知っていた外積を実際にプログラムを利用して求めることができたので数学的な理論のプログラムとして作る方法がわかるようになった。また、ネットにある外積計算機もまたこのような方法で作られたプログラムであることを知ってプログラムの仕組みついてわかるようになった。