計算機入門及び演習
C 言語レポート課題 関数

6321120
横溝尚也

提出日：12 月　9 日　（木）

1　レポート課題１

以下の関数を呼び出して返り値を出力する。
返り値が -1 の場合は出力しない。

関数

次の漸化式で定義される数列 xn があるとする。
$x_n = x_{n\ -\ 1} + arg1$
$x_1 = arg2$
この数列について、x_{arg3} を返す。
arg1、arg2、arg3 は 1 以上の整数値であり、引数として受け取る。
いずれかの引数が 1 以上ではない場合は エラーメッセージを出力して-1 を返す

2　アルゴリズムの説明

指定された関数内では以前やったように漸化式を一つ一つ計算して値を出していけばよい。またその関数には３つの引数を用いる。つまりメイン関数のなかでサブ関数の値を宣言することができるという事である。返すとはサブの関数で返り値として一つ宣言すれば、メイン関数内でそれを受け取ることが可能という事である。

3　プログラムの説明

```
1#include <stdio.h>

2int sequence (int arg1 , int arg2 , int arg3){ /*関数 sequence を宣言*/
3    int xn = arg2;
/*整数型変数 xn と i を宣言し xn を arg2 を代入*/
4    int i;

5    for (i = 1; i < 10000; i++){
/*くり返し文を利用して n が arg3 まで漸化式を計算する*/
6        if (i == arg3){
7            if ( arg1 >= 1 && arg2 >= 1 && arg3 >= 1){ /*漸化式の計算後返す値を条件分岐*/
8                return xn;
9            }else{
10                return -1;
11            }
```

```
12       }else{
13            xn = xn + arg1;
14        }
15    }
16}
17int main() {
18    int result;                              /*整数型変数 result,arg1,arg2,arg3 を宣言*/
19    int arg1 , arg2 , arg3;

20    arg1 = 5;                                /*arg1,arg2,arg3 にそれぞれ任意の整数を代入*/
21    arg2 = 3;
22    arg3 = 8;

23   result = sequence (arg1 ,arg2 ,arg3);
/*関数 sequence の返り値によって出力結果を条件分岐*/
24    if( result != -1){
25        printf("----------\n");
26        printf("arg1:%d\n" , arg1);
27        printf("arg2:%d\n" , arg2);
28        printf("arg3:%d\n" , arg3);
29        printf("----------\n");
30        printf("x%d:%d\n" , arg3 , result);

31    }else{
32    printf("----------\n");
33        printf("arg1:%d\n" , arg1);
34        printf("arg2:%d\n" , arg2);
35        printf("arg3:%d\n" , arg3);
36        printf("----------\n");
37        printf("Error: 引数が１以上の整数値ではない。\n");
38   }
39}
```

1から１６行目がサブの関数となる sequence を記述している。$x_n = x_{n-1} + arg1$ と $x_1 = arg2$ の２式の条件式から x_{arg3} の値を出力するようにプログラムしたのが３行目から１５行目である。５行目では漸化式で新たな値を求めるためにくり返し文を利用している。繰り返し回数が arg3 に達したのか否かで if 文を用い、条件分岐している。arg3 に達した場合、また更に if 文を用いて返り値の設定をしているのが６から１１行目である。arg3 に達していない場合、達するまで漸化式の計算をするようにプログラムしているのが１２行目から１５行目である。

１７行目以降にメイン関数のプログラムが記載されている。まずは１８行目から２２行目では sequence 関

数で使用しメイン関数でも利用する関数を宣言し、先ほど求めた返り値を使用するために result を宣言する。また、引き値とした arg1、arg2、arg3 に整数を代入した。２３行目で作成した sequence 関数を実行し、返り値の値で条件分岐を用いて処理内容を記述したのが２４行目から３８行目である。２４行目が返り値ー１でないときの処理、３１行目が返り値が―１の時の処理となっている。

4　実行結果

図１　実行結果 (1)

5　考察

　このプログラムが優れている点はメイン関数内で代入した arg1 arg3 の値を変更するだけで様々な漸化式を計算することができる点である。この課題だけではサブの関数内で代入した値を変更しても手間は変わらない

と感じるかもしれないがこの漸化式の計算を代入する値を変更して何度も結果を出力したいときにメイン関数で sequence を実行しその時に引き値を変えるだけで手間がかなり省けていく。

今までならば、出力内容を条件分岐にする際に if 文を用いてメイン関数内の条件式が長くなり簡潔でなくなることがある。しかし、その条件分岐をサブの関数で行い、メイン関数で返り値の値によって出力内容を決めているため、メイン関数がより簡潔になっている。

6　レポート課題２

以下の関数を呼び出す。

関数

0 以上 500 以下の数値 2 つと演算子 1 つを引数として受け取り、それらを使用した計算結果を出力する。
例えば、「5」「2.5」「/」を受け取った場合は「5 ÷ 2.5」の計算結果を出力する。
演算子は「+」「-」「*」「/」のいずれかとする。
以下のいずれかの条件を満たす場合、計算結果ではなく、それぞれに応じたエラーメッセージを出力する。
(1) 受け取った数値が 0 以上 500 以下ではない。
(2) 受け取った演算子が「+」「-」「*」「/」のいずれでもない。
(3) 計算がゼロ除算となる。

7　アルゴリズムの説明

数値二つと演算子一つを引数として受け取り、その情報から計算結果を出力すればよい。つまり、受け取った引数の値から新たに計算結果を格納する変数を用意してその変数を最後に出力すればよい。

その際、（１）～（３）に記載されている例外の場合も考慮しなければならない。この例外に当てはまるか当てはまらないのかは条件式を用い判断させればよい。

8　プログラムの説明

```
1    #include <stdio.h>

2int function (float num1 , float num2 , char operator ){
/*少数型変数 num1,num2 と文字型変数 operator を関数 function の引数として宣言*/
3    if ( num1 < 0 || num1 > 500 || num2 < 0 || num2 > 500 ){
    /*num1,num2 が０以上５００以下でない場合のエラーメッセージの出力処理*/
4        printf("Error: 数値が０以上５００以下ではない。\n");
5        printf("数値１:%f\n" , num1 );
6        printf("数値２:%f\n" , num2 );

7    }else if(operator !='+' && operator !='-' && operator !='*' && operator !='/'){
    /*演算子 operator が適切でないときのエラーメッセージの出力処理*/
8        printf("Error: 演算子が適切ではない。\n");
9        printf("演算子:%c\n" , operator);
```

```
10    }else if ( operator == '/' && num2 == 0){
    /*ゼロ除算となる場合のエラーメッセージの出力処理*/
11        printf("Error: 計算がゼロ除算となる。\n");
12        printf("%f%c%f\n" , num1 , operator , num2);

13    }else{
14        float answer;                 /*正常に計算できる場合の計算処理*/
15        if (operator == '+'){         /*それぞれの演算子の時の計算結果の代入*/
16            answer = num1 + num2;
17        }else if (operator == '-'){
18            answer = num1 - num2;
19        }else if (operator == '*'){
20            answer = num1 * num2;
21        }else{
22            answer = num1 / num2;
23        }

242        printf("%f%c%f=%f\n" , num1 , num2 , operator , answer );
25    }
26}
27int main() {
28    float num1 , num2 , answer;  29        /*関数 function 内のプログラムを実行するための
変数の宣言*/
30    char operator;

31    num1 = 1.500000;                    /*任意の数値２つと演算子１つの代入*/
32    num2 = 2.500000;
33    operator = '+';

34    function( num1 , num2 , operator); /*関数 function のプログラムの実行*/

35}
```

２から２６行目までが関数 function の記述である。３から６行目で（１）の場合のエラーメッセージを出力する際のプログラム、７から９行目が（２）１０から１２行目が（３）を記述している。

１３から２５行目では正常に引数を受け取ったときの計算結果を出力するプログラムである。またその中で１４から２３行目では４つの演算子のうちそれぞれの演算子であるときの処理を記述している。

２７行目以降はメイン関数である。まずは関数 function でも使用し、メイン関数内でも使用するための変数を宣言し、二つの数値、一つの演算子を代入している。３４行目では前半で作成した関数を実行している。

9　考察

今回の課題で作成した関数 function を作成し、その後メイン関数でその関数を実行するという全体的な流れとなった。メイン関数内ですべてのプログラムをするよりも優れている点は、いくつかの四則演算をしたい時などに引数を代入し、関数 function を実行するだけで何度もプログラムを実行できる。一方でメイン関数内にすべてのプログラムを記述した場合、四則演算をするたびに同じプログラムを書く必要があり、プログラムがとても長くなる。よって新たな関数にプログラムすることでかなり簡潔になった。

一つ注意しなければならないのが引数の変数の型である。整数とは限定されていないので二つの数値は少数型変数で宣言し、演算子は文字型変数として宣言する必要がある。

このプログラムにはまだ改善点があるとこの考察をしているときに感じた。それは計算結果を新たな変数 answer に格納するときのプログラムである。４つの演算子それぞれで if 文を用いて条件分岐し、answer に格納している。この時 if 文よりも switch 文を使用した方が今回の場合は簡潔になると考えた。

10　実行結果

図 2　実行結果 (2)

11　感想

この関数のレポートを通してメイン関数をより簡潔にし、プログラムを見やすくするための方法を学び、その大切さを理解できた。

今までは関数は一つであったので土の関数をプログラムしているのかなど考えることがなかったけど、自分がプログラミングしているときに今全体のプログラムのうちどの関数を記述しているのかしっかりと考えるようになった。