

C 言語レポート課題 配列

6321120

横溝尚也

提出日：11月 23日 (火)

1 レポート課題 1

次の漸化式で定義される数列 x_n について x_3 から x_7 を逆順で出力する。

$$x_1 = 300$$

$$x_n = x_{n-1} \div 2$$

2 アルゴリズムの説明

第3項から第7項までの値を逆順に出力するので、順に出力するときのようにくり返し文だけを使って出力することは難しい。よってまず初めに初項から第7項までの値を配列に格納する。そののちに一つ一つの項の値を出力していく。

3 プログラムの説明

```
1#include <stdio.h>

2int main(){

3    float array[7];          /*少数型変数として配列 array[] を格納*/
4    int i;                   /*整数型変数 i の宣言*/
5    float xn = 300;          /*少数型変数 xn の宣言と代入*/

6    for(i = 0; i < 7; i++){  /*くり返し文の利用*/
7        array[i] = xn;      /*配列 array[] を xn の値で格納する*/
8        xn = xn / 2;        /*xn を 2 で割った値を新たに xn とする*/
9    }

10   for(i = 7; i > 2 ;i--){  /*くり返し文の利用*/
11       printf("x%d = %f\n" , i , array[i-1] ); /*格納した配列を出力する*/
12   }
13}
```

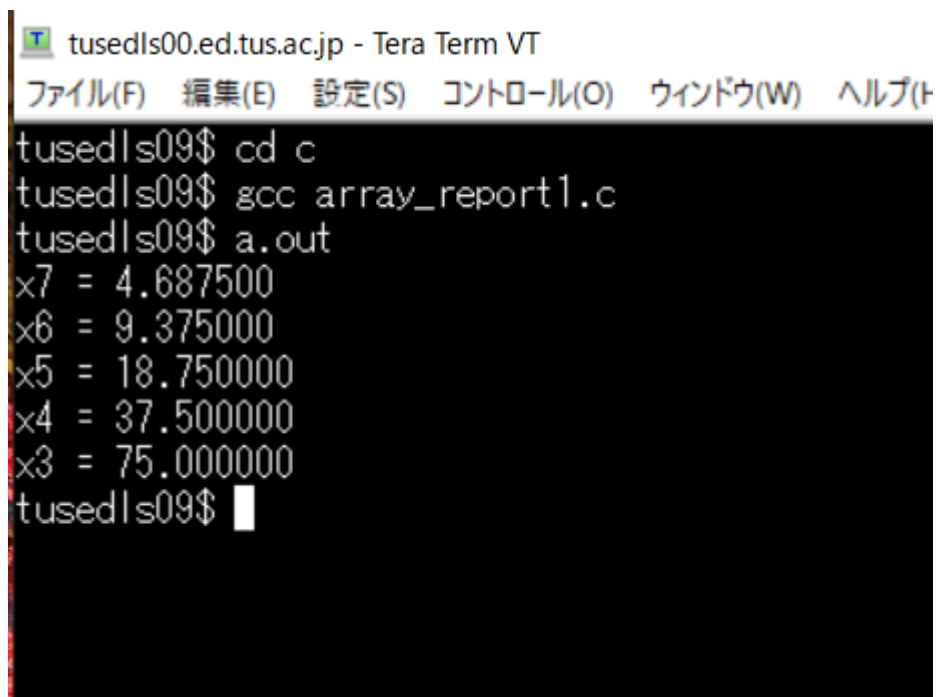
3～5 行目では変数の格納を行っている。漸化式を解いた項の値を格納するための配列を3行目で宣言。4行目で宣言した i はくり返し文利用する際に繰返し範囲を設定するための変数。5行目の x_n は少数になりえるので少数型変数として格納している。

6～9行目ではくり返し文を利用して簡潔に数列の項の値を格納している。10～16行目では主に格納した項の出力を行っている。逆順で出力するという指示があるので i を7から初めて i が1ずつ下がっていくよう $i = 3$ までに範囲を指定している。

4 考察

以前の課題であったように漸化式の項を順に出力するのならくり返し文を一つ使うだけで第 k 項の値を計算し、さらに出力することができた。この場合わざわざ配列に格納する必要もなく簡潔にプログラムすることができた。しかし、今回の課題では逆順に項の値を出力しなければならない。この場合、計算して得られた値をどこかに保存しておいて一番最初に出力する項まで計算ができれば、出力に入るという流れになる。これを可能にするのが配列である。配列に値を格納したのちに出力する。またこの時多くの項を配列に格納して、出力するのはかなり手間がかかるので、配列への格納と出力の2段階でともに繰り返し文を使えばプログラムを簡潔にできると考えた。

5 実行結果



```
tusedls00.ed.tus.ac.jp - Tera Term VT
ファイル(F) 編集(E) 設定(S) コントロール(O) ウインドウ(W) ヘルプ(H)
tusedls09$ cd c
tusedls09$ gcc array_report1.c
tusedls09$ a.out
x7 = 4.687500
x6 = 9.375000
x5 = 18.750000
x4 = 37.500000
x3 = 75.000000
tusedls09$
```

図1 実行結果 (1)

1 レポート課題2

テストの点数と受験者の番号の表が次のようになっている。

| | | | | | | | | | | | | | |
|-------|----|----|----|----|----|----|----|----|----|----|----|----|----|
| 受験者番号 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 |
| 点数 | 10 | 40 | 90 | 70 | 50 | 80 | 20 | 90 | 60 | 50 | 70 | 60 | 0 |

このテストについて、60点未満だった受験者の番号を表示する。このプログラムを作成する際は、以下のデータをコピーして使用しても良い。

{10, 40, 90, 70, 50, 80, 20, 90, 60, 50, 70, 60, 0}

2 アルゴリズムの説明

全受験者の中から60点未満である受験者番号を出力する、つまりコンピュータに受験者の点数をいったん認識させてからその点数が60点未満かどうかを判別させればよい。

3 プログラムの説明

```
1#include <stdio.h>

2int main(){
3    int array[13] = {10, 40, 90, 70, 50, 80, 20, 90, 60, 50, 70, 60, 0};
/*配列 array に受験者のテストの点数をそれぞれ格納する*/
4    int i;
/*整数型変数 i の宣言*/

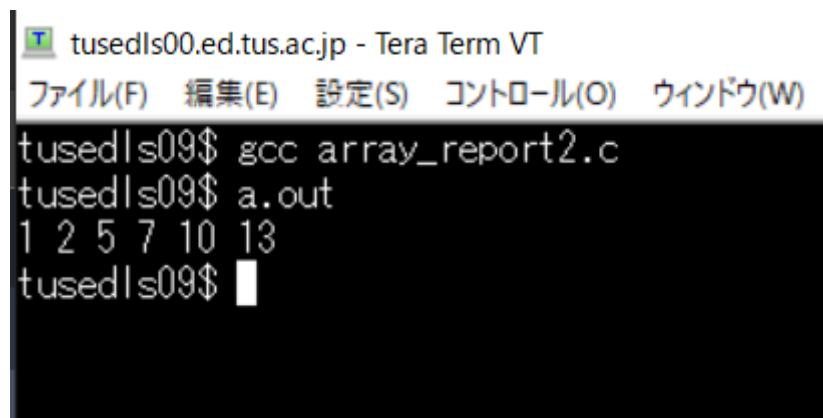
5    for (i = 0; i < 13; i++){
/*くり返し文を利用*/
6        if(array[i] < 60){
/*array[i] が60未満の時の場合*/
7            printf("%d ", i+1);
/*条件を満たすときに受験者番号を表示*/

8        }else{
9        }
10    }
11    printf("\n");
/*改行*/

12}
```

3行目では配列 array を宣言しそれに与えられた全受験者の点数を格納している。5行目以降では格納した点数を出力するか否かを判別している。6行目で if 文で条件式を格納した点数が60未満であるとしている。この条件式を満たす受験者の点数は出力を命令しているのが7行目である。出力する際に受験者番号を出力するので7行目の引数に i+1 と命令している。

4 実行結果



```
tusedls00.ed.tus.ac.jp - Tera Term VT
ファイル(F) 編集(E) 設定(S) コントロール(O) ウィンドウ(W)
tusedls09$ gcc array_report2.c
tusedls09$ a.out
1 2 5 7 10 13
tusedls09$
```

図2 実行結果 (2)

5 考察

受験者の点数やクラスの身長など役割が同じ変数を複数扱いたいとき、配列に格納するのが最適である。よって今回も配列に格納する。ここで配列を使ったもう一つの理由が、配列の添え字と受験者番号の対応である。今回の場合受験者が13人であることから配列の要素数を13とし、受験番号順に配列に格納すれば（配列の添え字）+1が受験者番号に対応している。+1する理由は添え字が0から始まり、（要素数）-1が最後の添え字である。

1 レポート課題3

50 名の受験者を 5 つの組に分けてテストを実施した結果、以下のようになった。

| | | | | | | | | | | |
|-------|----|----|----|----|----|----|----|----|----|-----|
| 1 組 | | | | | | | | | | |
| 受験者番号 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 |
| 点数 | 83 | 0 | 68 | 39 | 84 | 26 | 4 | 18 | 18 | 89 |
| 2 組 | | | | | | | | | | |
| 受験者番号 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 |
| 点数 | 72 | 17 | 17 | 6 | 5 | 28 | 24 | 94 | 8 | 98 |
| 3 組 | | | | | | | | | | |
| 受験者番号 | 30 | 31 | 32 | 33 | 34 | 35 | 36 | 37 | 38 | 39 |
| 点数 | 45 | 63 | 44 | 78 | 72 | 38 | 31 | 80 | 7 | 78 |
| 4 組 | | | | | | | | | | |
| 受験者番号 | 40 | 41 | 42 | 43 | 44 | 45 | 46 | 47 | 48 | 49 |
| 点数 | 56 | 57 | 44 | 23 | 96 | 95 | 16 | 66 | 12 | 0 |
| 5 組 | | | | | | | | | | |
| 受験者番号 | 50 | 51 | 52 | 53 | 54 | 55 | 56 | 57 | 58 | 59 |
| 点数 | 54 | 50 | 85 | 72 | 56 | 90 | 66 | 47 | 49 | 100 |

このテストについて、以下を表示する

(点数が最も高い受験者が複数人いる場合は考慮しなくて良い)。

- (1) 全受験者の平均点
- (2) 各組の平均点
- (3) 点数が最も高い受験者の点数と所属する組と番号

このプログラムを作成する際は、以下のデータをコピーして使用しても良い。

{83, 0, 68, 39, 84, 26, 4, 18, 18, 89}

{72, 17, 17, 6, 5, 28, 24, 94, 8, 98}

{45, 63, 44, 78, 72, 38, 31, 80, 7, 78}

{56, 57, 44, 23, 96, 95, 16, 66, 12, 0}

{54, 50, 85, 72, 56, 90, 66, 47, 49, 100}

2 アルゴリズムの説明

平均点とはデータの値をすべて足し合わせ、それをデータ数で割った値である。よって各組の平均点を求めるには各組の受験者の点数を足し合わせそれを 10 で割ればよい。全受験者の平均点も同様である。

(3) では最も点数の高い生徒のデータがどれか求めればよい。基準となるデータを一つ選びそれとほかのデータの点数どちらが大きいか比較していく。基準となるデータよりも小さければ基準はそのまま、大きければ基準となるデータをそのデータに変更する。これを 50 回比較し行けば最も高い点数のデータを抽出でき

る。

3 プログラムの説明

```
1#include <stdio.h>

2int main(){
3    int array[5][10] = { {83, 0, 68, 39, 84, 26, 4, 18, 18, 89} ,
    /*配列 array に全受験者の点数を格納*/
4        {72, 17, 17, 6, 5, 28, 24, 94, 8, 98} ,
5        {45, 63, 44, 78, 72, 38, 31, 80, 7, 78} ,
6        {56, 57, 44, 23, 96, 95, 16, 66, 12, 0} ,
7        {54, 50, 85, 72, 56, 90, 66, 47, 49, 100} ,
8    };
9    int i , j ;
    /*整数型変数 i,j の宣言*/
10    float score;
    /*少数型変数 score の宣言*/
11    int number , class;
    /*整数型変数 number,class の宣言*/

12    for(i = 0;i < 5;i++){
    /*くり返し文を2回利用して全受験者の点数を足す*/
13        for(j = 0; j < 10; j++){
14            score = score + array[i][j];
15        }
16    }
17    score = score / 5 / 10;
    /*点数をすべて足し合わせたものを人数で割る*/
18    printf("---全受験者の平均点---\n%f\n" , score);
    /*平均点の出力*/
19    score = 0;
    /*score をリセット*/

20    printf("----各組の平均点---\n");
21    for(i = 0; i < 5;i++){
    /*各クラス分以下を繰り返す*/
22        for(j = 0; j < 10;j++){
```

```

        /*クラス全員の点数をくり返し文を使って足す*/
23     score = score + array[i][j];
    }
24     score = score / 10;
    /*クラスの人数で割る*/
25     printf("%d組:%f点\n" , i+1 , score );
    /*クラスの平均点を出力*/
26     score = 0;
    /*score をリセット*/
27 }

28 printf("---最高得点---\n");
29 for(i = 0;i < 5;i++){
    /*2 回の繰返し文を使って全受験者の点数が最高点であるのかを調べる*/
30     for(j = 0;j < 10;j++){
31         if(array[i][j] > score){
            /*最高点であるのか条件式を用いて調べる*/
32             score = array[i][j];
            /*最高点を更新*/
33             number = i * 10 + 10 + j;
            /*現時点での最高点である人の受験番号を更新*/
34             class = i + 1;
35         }else{
36         }
37     }
38 }

39 printf("点数:%f点\n" , score);
    /*最高点、番号、所属をそれぞれ出力*/
40 printf("番号:%d\n" , number );
41 printf("所属:%d\n" , class);

42}

```

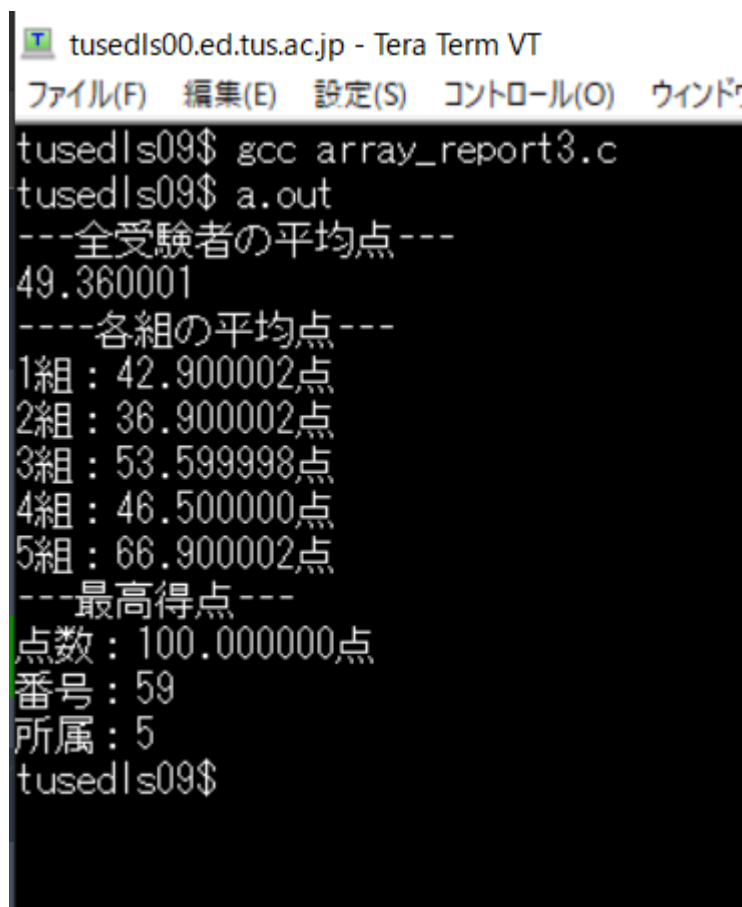
1 行目から 11 行目までは全受験者の点数を配列に格納し、整数型変数を宣言している。12 行目から 19 行目では全受験者の平均点を出力するためのプログラムである。少数型変数 score に格納したすべての点数を足し合わせているのが 14 行目であり、17 行目で受験人数 50 で割っている。

次に 20 行目から 27 行目では各組の平均点を出力するためのプログラムである。21 行目のくり返し文では 1 組から 5 組までの平均点を求める作業では同じプログラムであるのでくり返し文を利用して 1 クラス分の平

均点を求める作業で済ませている。

28行目から41行目では最高点である受験者の出力するためのプログラムである。暫定最高点である点数、受験番号、所属クラスを変数に格納して、ifを用いて暫定最高点よりも新たな受験者の点数が高いのか低いのかで条件分岐している。

4 実行結果



```
tusedls00.ed.tus.ac.jp - Tera Term VT
ファイル(F) 編集(E) 設定(S) コントロール(O) ウィンドウ(W)
tusedls09$ gcc array_report3.c
tusedls09$ a.out
---全受験者の平均点---
49.360001
----各組の平均点---
1組: 42.900002点
2組: 36.900002点
3組: 53.599998点
4組: 46.500000点
5組: 66.900002点
---最高得点---
点数: 100.000000点
番号: 59
所属: 5
tusedls09$
```

図3 実行結果 (3)

5 考察

配列と繰り返し文を利用することで数多くのデータを一度に扱うことができ今までよりもプログラムがかなり簡潔になったと思う。また、自分のプログラムでは(1)の全受験者の平均点を出力した後に、(2)の各組の平均点を出力したが、先に(2)をプログラムすることで各組の平均点に人数10をかけ、平均点を足し合わせれば全受験者の合計点が求まるのでプログラムが簡潔に書けるのでより良いプログラムになると思った。

1 レポート課題 4

以下の計算結果を配列に格納する。

$$a+5 \times b-c$$

a と b と c は 0~9 の任意の自然数であり、これらを指定すると計算結果を出力する

2 アルゴリズムの説明

与えられた計算は 3 つの変数を用いた四則計算である。よってこの計算結果を配列に格納するには 3 次元配列に格納する必要がある。そして計算結果を格納した後に任意の自然数を 3 つ宣言し、配列から計算結果を出力すればよい。

3 プログラムの説明

```
1#include <stdio.h>

2int main(){
3    int array[10][10][10];                /*配列 array を宣言する*/
4    int i, j, k;                          /*整数型変数,i,j,k を宣言*/

5    for(i=0;i < 10;i++){                  /*繰り返し文を 3 回利用し配列に格納する*/
6        for(j=0;j < 10;j++){
7            for(k=0;k < 10;k++){
8                array[i][j][k] =(i+1)+5*(j+1)-k-1;
9            }
10        }
11    }

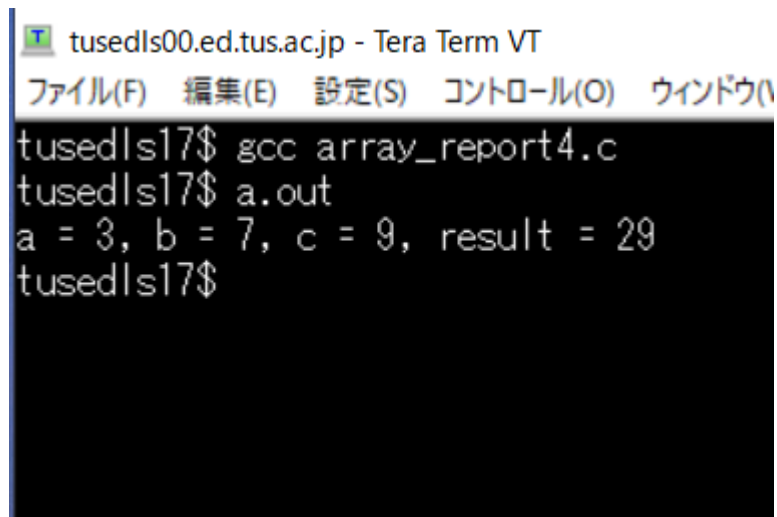
12    int a = 3;                            /*任意の自然数 a,b,c を宣言*/
13    int b = 7;
14    int c = 9;
15    printf("a = %d, b = %d, c = %d, result = %d\n" ,a,b,c,array[a-1][b-1][c-1]);
    /*計算結果の出力*/

16}
```

3 行目では 3 次元配列を宣言している。5 行目から 11 行目では 3 次元配列にくり返し文を 3 つ利用して配列に計算結果をすべて格納している。最後に a,b,c の値を代入して、その代入した値から計算結果を配列から

とってきて出力している。

4 実行結果



```
tusedls00.ed.tus.ac.jp - Tera Term VT
ファイル(F) 編集(E) 設定(S) コントロール(O) ウィンドウ(W)
tusedls17$ gcc array_report4.c
tusedls17$ a.out
a = 3, b = 7, c = 9, result = 29
tusedls17$
```

図 4 実行結果 (4)

5 考察

この課題では他次元配列の利便性をうまく利用した。計算結果を格納するときに変数が3つ存在するので3次元配列を使用し、各要素数を10にする。すると0～9の任意の自然数を指定した時の計算結果を一つの配列にすべて格納することができる。出力する際に引数を用いて配列から計算結果を持ってくればよい。今回の課題では使用しなかったが乱数を使うことによってプログラムの幅も広がると考えた。たとえば0～9の自然数のどれかを乱数を用いて選び出しそれを a,b,c として宣言することもできる。

6 感想

今回の課題では配列に格納することによって膨大な量のデータを同時に扱うことができ、プログラムの幅がかなり広がったと思う。