Structure and Analysis of Algorithm

Assignment-2

Group-2 (IEC2021005-IEC2021009)

Gaurav Sahay	Ansh Kothari	Sonali Gupta	Pragya Pal	Ayush Chandra
(IEC2021005)	(IEC2021006)	(IEC2021007)	(IEC2021008)	(IEC2021009)

Introduction to Programming

INDIAN INSTITUTE OF INFORMATION TECHNOLOGY, ALLAHABAD

***Abstract*— A record contains name of farmer, his age, number of acres of land that he has cultivated and the average crop that he has produced in each crop. Create an array of structure to hold records of 15 such farmers and then write a program to read these records and arrange them in ascending order by average of crops. Use the qsort( ) standard library function and collect crop data per each acre.**

I. INTRODUCTION

Given a record which contains name of farmer, age, number of acres of land and the average crop produced per acre. We have to create an array of structure which contains the record of some farmers and ascend the data by average of crops produced in a acre. The concept of qsort and pointers will make our task easier. Also, the concept of array of structures is also helpful in case of ascending all the records.

An array of structures is a collection of different datatype variables, grouped together under a single name.

A pointer is a variable whose value is the address of another variable, i.e. direct address of the memory location. qsort is a C library function that uses a quick sort algorithm to sort an array.

II. ALGORITHM DESIGN

- We initiated the program by creating a structure farmer using typedef having members as name to store name of farmer ,age to store age of farmer, noa to store number of acres of farmer and avg to store average crop produced in each acre.
- Here n stores the number of farmers and we have created an array of structure farmer f[n] of size n.
- Int compare function is created having two pointers as parameter with void type which is further typecasted into farmer and returns the difference of average crop produced by farmers.This function is to be used by qsort() to arrange the input of farmers according to ascending order.
- For loop is executed to take the input of all farmers with their details like name , age number of acres and average crop produced.
- Qsort() function is used having array name ,size of array , sizeof(farmer) function and compare function as parameter.
- After the implementation of qsort() function , all the data of farmers will be set in ascending order of average crops.
- For loop is executed to print the data of all farmers in ascending order of average crop produced.

III. ALGORITHM AND ILLUSTRATION

```
BEGIN
STRUCTURE farmer{
    DATATYPES
       CHARACTER ARRAY name[30]
       INTEGER AGE
       FLOAT noa
       FLOAT avg }
INTEGER n=15
ARRAY  OF STRUCTURE farmer f[n]
FUNCTION compare with RETURNTYPE INTEGER
(PARAMETERS CONSTANT VOID *pa, CONSTANT VOID *pb){
    farmer POINTER*p1=(farmer*)pa
    farmer POINTER*p2=(farmer*)pb
    RETURN p1->avg - p2->avg
}
```

```
FOR INTEGER x=0,x<=n-1,x=x+1
    DISPLAY Enter name of farmer %d(farmer number)
    INPUT f[x].name
    DISPLAY Enter age of farmer %d(farmer number)
    INPUT f[x].age
    DISPLAY Enter number of acres of farmer %d(farmer number)
    INPUT f[x].noa
    DISPLAY Enter average crop produced in
    each acre of farmer %d(farmer number)
    INPUT f[x].avg
ENDFOR

Declare FUNCTION qsort(ARGUMENTS f(ARRAY of STRUCTURE farmer),
n(size of array f),sizeof(FLOAT DATATYPE),FUNCTION compare)

FOR INTEGER x=0,x<=n-1,x=x+1
     DISPLAY f[x].name,f[x].age,f[x].noa,f[x].avg
ENDFOR
```

Input:

```
Enter name of farmer 1 :Gaurav
Enter age of farmer 1 :18
Enter number of acres of farmer 1 :321
Enter average crop produced in each acre of farmer 1 :543
Enter name of farmer 2 :Ansh
Enter age of farmer 2 :18
Enter number of acres of farmer 2 :543
Enter average crop produced in each acre of farmer 2 :432
Enter name of farmer 3 :Sonali
Enter age of farmer 3 :18
Enter number of acres of farmer 3 :324
Enter average crop produced in each acre of farmer 3 :543
Enter name of farmer 4 :Pragya
Enter age of farmer 4 :18
Enter number of acres of farmer 4 :532
Enter average crop produced in each acre of farmer 4 :765
Enter name of farmer 5 :Ayush
Enter age of farmer 5 :543
Enter number of acres of farmer 5 :432
Enter average crop produced in each acre of farmer 5 :654
Enter name of farmer 6 :qwerty
Enter age of farmer 6 :18
Enter number of acres of farmer 6 :432
Enter average crop produced in each acre of farmer 6 :543
Enter name of farmer 7 :rety
Enter age of farmer 7 :18
Enter number of acres of farmer 7 :678
Enter average crop produced in each acre of farmer 7 :432
Enter name of farmer 8 :Beta
Enter age of farmer 8 :18
Enter number of acres of farmer 8 :432
Enter average crop produced in each acre of farmer 8 :554
Enter name of farmer 9 :Gamma
Enter age of farmer 9 :34
Enter number of acres of farmer 9 :748
Enter name of farmer 9 :Gamma
Enter age of farmer 9 :34
Enter number of acres of farmer 9 :748
Enter average crop produced in each acre of farmer 9 :785
Enter name of farmer 10 :Mrfel
Enter age of farmer 10 :54
Enter number of acres of farmer 10 :656
Enter average crop produced in each acre of farmer 10 :543
Enter name of farmer 11 :Reshima
Enter age of farmer 11 :54
Enter number of acres of farmer 11 :653
Enter average crop produced in each acre of farmer 11 :543
Enter name of farmer 12 :Serry
Enter age of farmer 12 :54
Enter number of acres of farmer 12 :657
Enter average crop produced in each acre of farmer 12 :765
Enter name of farmer 13 :YU
Enter age of farmer 13 :53
Enter number of acres of farmer 13 :432
Enter average crop produced in each acre of farmer 13 :654
Enter name of farmer 14 :Welsi
Enter age of farmer 14 :51
Enter number of acres of farmer 14 :543
Enter average crop produced in each acre of farmer 14 :654
Enter name of farmer 15 :Tim
Enter age of farmer 15 :65
Enter number of acres of farmer 15 :535
Enter average crop produced in each acre of farmer 15 :654
```

OUTPUT:

```
Ansh  18  543.000000  432.000000
rety  18  678.000000  432.000000
Gaurav  18  321.000000  543.000000
Sonali  18  324.000000  543.000000
qwerty  18  432.000000  543.000000
Mrfel  54  656.000000  543.000000
Reshima  54  653.000000  543.000000
Beta  18  432.000000  554.000000
Ayush  543  432.000000  654.000000
YU  53  432.000000  654.000000
Welsi  51  543.000000  654.000000
Tim  65  535.000000  654.000000
Pragya  18  532.000000  765.000000
Serry  54  657.000000  765.000000
Gamma  34  748.000000  785.000000
```

IV. ALGORITHM ANALYSIS

Time complexity:

The execution time of above algorithm is calculated as **0.002372seconds**.
It is calculated using following syntax:

```
#include <time.h>

clock_t start, end;
double cpu_time_used;

start = clock();
... /* Do the work. */
end = clock();
cpu_time_used = ((double) (end - start)) / CLOCKS_PER_SEC;
```

```
Enter average crop produced in each acre of farmer 11 :56
Enter name of farmer 12 :sss
Enter age of farmer 12 :34
Enter number of acres of farmer 12 :45
Enter average crop produced in each acre of farmer 12 :67
Enter name of farmer 13 :hh
Enter age of farmer 13 :34
Enter number of acres of farmer 13 :45
Enter average crop produced in each acre of farmer 13 :88
Enter name of farmer 14 :ff
Enter age of farmer 14 :12
Enter number of acres of farmer 14 :34
Enter average crop produced in each acre of farmer 14 :90
Enter name of farmer 15 :rr
Enter age of farmer 15 :34
Enter number of acres of farmer 15 :56
Enter average crop produced in each acre of farmer 15 :100
w  23  34.000000  45.000000
f  12  34.000000  47.000000
ayush  20  1200.000000  50.000000
c  23  34.000000  56.000000
h  23  456.000000  56.000000
l  34  45.000000  56.000000
d  13  45.000000  66.000000
sss  34  45.000000  67.000000
e  23  44.000000  88.000000
hh  34  45.000000  88.000000
a  12  34.000000  90.000000
ff  12  34.000000  90.000000
q  23  34.000000  99.000000
rr  34  56.000000  100.000000
qwerty  23  34.000000  120.000000
time taken = 0.002372

...Program finished with exit code 0
Press ENTER to exit console.
```

V. CONCLUSION

- **We may conclude that structure gives us a powerful way to store data of many types in a single object and accessing the data is also very easy.**
- **Array of structures is very useful when we have to store data of more than one entity.**
- **qsort() function is a great tool to arrange elements of array in ascending or descending according the requirement.**

VI. REFERENCES

- Array of Structures vs. Array within a Structure in C/C++ - GeeksforGeeks
- Comparator function of qsort() in C – GeeksforGeeks
- "**Let us C**" Book by Yashvant Kanetkar

APPENDIX

Code for implementation of this paper is given below:

```
#include <stdio.h>
#include <stdlib.h>

int main()
{
    typedef struct
    {
        char name[30];
        int age;
        float noa;
        float avg;
    }farmer;
    int n=15;
    farmer f[n];
    int compare(const void *pa, const void *pb){
        farmer *p1 = (farmer *)pa;
        farmer *p2 = (farmer *)pb;
        return (p1->avg - p2->avg);
    }


    for (int x = 0; x <= n - 1; x++)
    {
        printf("Enter name of farmer %d :", (x + 1));
        scanf("%s", (f[x].name));
        printf("Enter age of farmer %d :", (x + 1));
        scanf("%d", &(f[x].age));
        printf("Enter number of acres of farmer %d :", (x + 1));
        scanf("%f", &(f[x].noa));
        printf("Enter average crop produced in each acre of farmer %d :", (x + 1));
        scanf("%f", &(f[x].avg));
    }
    qsort(f, n, sizeof(farmer), compare);
    for (int x = 0; x <= n - 1; x++)
    {
        printf("%s %d %f %f \n", f[x].name, f[x].age, f[x].noa, f[x].avg);
    }
}
```

Listing 1. Code for this paper