STORING AND SORTING RECORDS OF 20 CRICKETERS

ITP Assignment-2

TABLE OF CONTENTS

PROBLEM STATEMENT

INTRODUCTION

PROCEDURE

TABLE OF CONTENTS

PROBLEM STATEMENT

Given a record containing the Name, Age, No. of tests played and the average runs in each test. Create an array of structure to hold records of 20 such cricketer and then write a program to read these records and arrange them in ascending order by average runs.

INTRODUCTION

We are given the records of 20 cricketers containing the Name, Age, No. of tests played and the average runs in each test. We will be using structure to store the information of the cricketers and qsort( ) to sort the cricketers by their average runs. For loops are also used to store and input the data in structure.

PROCEDURE

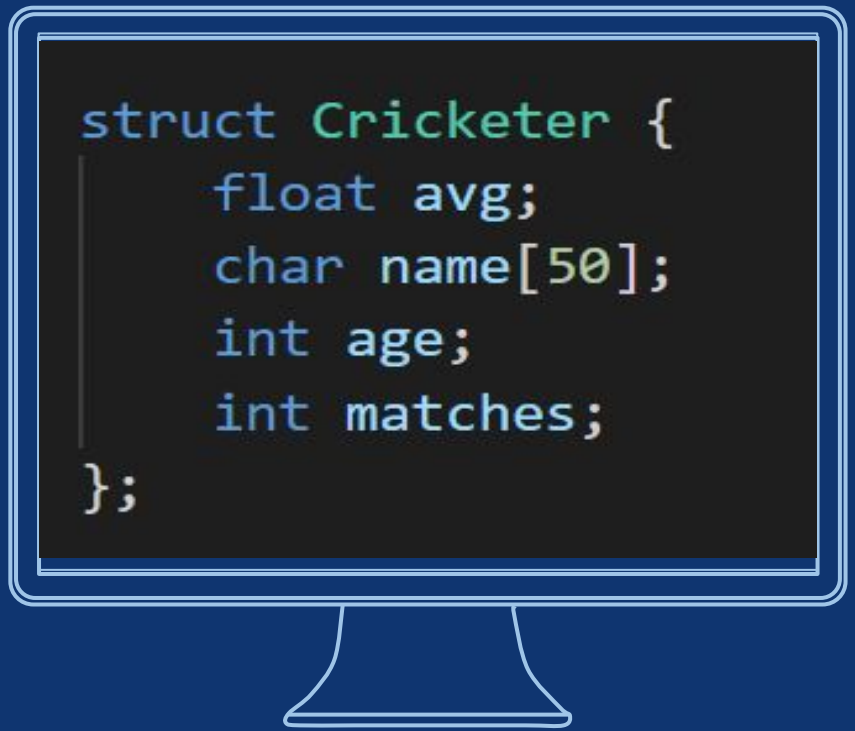
1. First we declare a structure named ‘Cricketer’ and initialize Average, Name, Age and Number of matches as members.
2. The next step would be to take the input from the user and store it in the array. We would be using the array of structures to implement this.
3. Now, declare a function named compare which would act as the comparator for the qsort() function. It takes two parameters as inputs and returns a value based on the relative order of the parameters.
4. Now use the qsort() function on the array of structures. This will sort the cricketers in the ascending order of their average runs.
5. The final step would be to print the required output.

ALGORITHM

Explanation of code

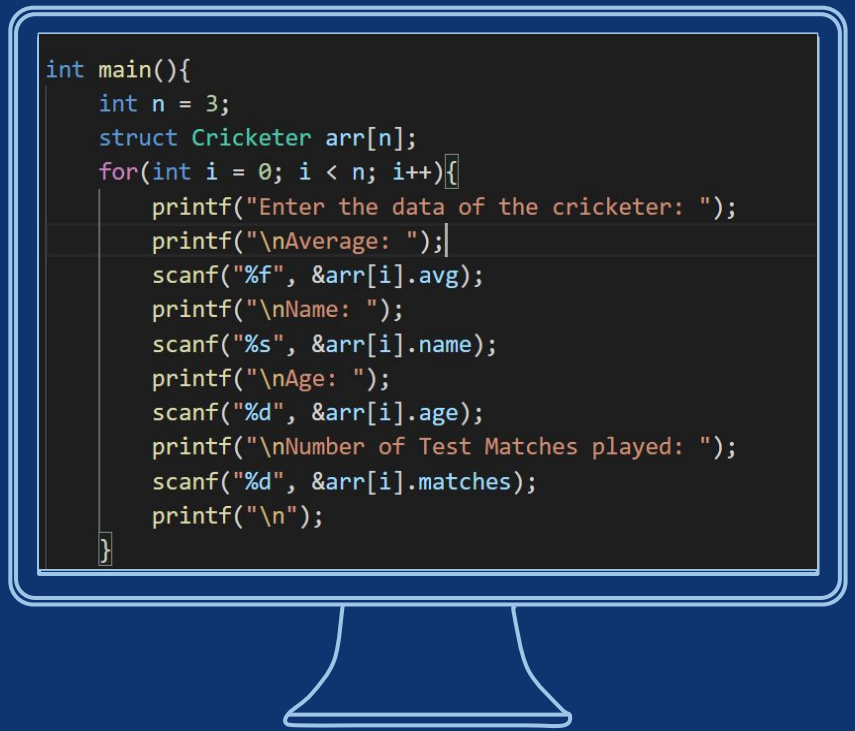
STEP 1

Declaring a structure and adding members inside it. This can be done as shown in the snippet.



STEP 2

Now we take input from the user and store it the array of structure, ‘arr’.



```c
int main(){
    int n = 3;
    struct Cricketer arr[n];
    for(int i = 0; i < n; i++){
        printf("Enter the data of the cricketer: ");
        printf("\nAverage: ");
        scanf("%f", &arr[i].avg);
        printf("\nName: ");
        scanf("%s", &arr[i].name);
        printf("\nAge: ");
        scanf("%d", &arr[i].age);
        printf("\nNumber of Test Matches played: ");
        scanf("%d", &arr[i].matches);
        printf("\n");
    }
```

STEP 3

Now we need a comparator function which gives an output based on the relative order of input parameters. As shown in the snippet, the function returns 1 if the first parameter is greater than the second. Else it returns -1. This helps in sorting the cricketers by taking each of the them inside the comparator function.

STEP 4

```
qsort(arr, n, sizeof(struct Cricketer), compare);
```

Now, the qsort( ) function is implemented as shown in the snippet. It calls the comparator function which decides the order.

STEP 5

The final step would be to print the required output which can be accomplished as shown.

```
printf("Cricketers arranged in ascending order by their Average: \n\n");
for(int i = 0; i < n; i++){
    printf("%s\t %f\n", arr[i].name, arr[i].avg);
}
```

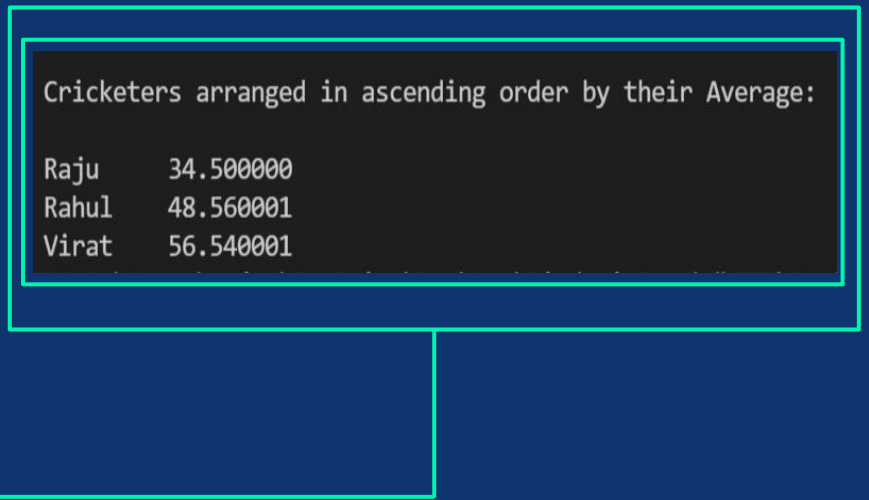
SAMPLE INPUT

This is an illustration to show how the code takes the input. The input is taken for 3 cricketers. We can do the same for 20 cricketers just by changing the size of the array.

```
Enter the data of the cricketer:
Average: 34.5

Name: Raju

Age: 34

Number of Test Matches played: 56

Enter the data of the cricketer:
Average: 56.54

Name: Virat

Age: 26

Number of Test Matches played: 78

Enter the data of the cricketer:
Average: 48.56

Name: Rahul

Age: 31

Number of Test Matches played: 45
```

CODE OUTPUT

The sample output which is the ascending order of cricketers based on their Average!



REFERENCES

[1] https://www.geeksforgeeks.org/structures-c/

[2] https://www.geeksforgeeks.org/comparator-function-of-qsort-in-c/

Thank you!

Submitted by:-

IEC2021050
IEC2021051
IEC2021052
IEC2021053