**Batch: A2 Roll No.: 16010121045**

**Experiment / assignment / tutorial No. 6**

**Grade: AA / AB / BB / BC / CC / CD /DD**

**Signature of the Staff In-charge with date**

|  |
| --- |
| **TITLE:**  Array of Structures. |

**AIM:** Program to declare an array of structure `players` having data members (name, total matches played, best bowling figure). Program should do the following operations using functions.

1. **Insert Minimum 5 player data in array of structure**
2. **Sort and display this data in descending order of their best bowling figure (if wickets are same then consider less run conceded as priority) and in proper tabular form**
3. **Delete the data for any one player.**
4. **Search for a particular player using its name.**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Expected OUTCOME of Experiment:**

CO3. Illustrate the use of derived and structured datatypes such as arrays, strings, structures and unions.

CO4. Design modular programs using functions and demonstrate the concept of pointers and file handling.

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Books/ Journals/ Websites referred:**

1. Programming in C, second edition, Pradeep Dey and Manas Ghosh, Oxford University Press.
2. Programming in ANSI C, fifth edition, E Balagurusamy, Tata McGraw Hill.
3. Introduction to programming and problem solving , G. Michael Schneider ,Wiley India edition.
4. [**http://cse.iitkgp.ac.in/~rkumar/pds-vlab/**](http://cse.iitkgp.ac.in/~rkumar/pds-vlab/)

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Problem Definition:**

Create an array of structure ‘players’ which store information about multiple players having different data members such as name, total matches played, best bowling figure.

Program should read choice from the user and perform following function:

Choice 1: Insert data in array of structure.

Choice 2: Sort and Display

Choice 3: Delete a player

Choice 4: Traverse and search a player with given name.

**Algorithm:**

**Void main()**

1. Start
2. Declare a structure **players** housing variables **name[20]** (char), **matches** (int), **bowfig** (int)
3. Declare an array of 5 players (**arr[5]**) as **global variable**
4. Ask the user to insert the data of 5 players in the array of structure
5. Declare tt=1 and size=5.
6. While(tt == 1)
7. Declare num
8. Display the menu.
9. Read the value in num
10. if (num==1)
11. **sort()**
12. **display(size)**
13. Else if(num==2)
14. Declare s
15. Ask the user to enter the player name whose data is to be deleted
16. Read the data in s
17. Store the returned value of **seach(s,size)** in n
18. If(n!=-1)
19. **Delete(n)**
20. Size=size-1
21. Display that player is deleted
22. Else
23. Display player not found
24. Else if(num==3)
25. Declare name
26. Ask the user to enter the player to be searched
27. Read name
28. Store the returned value of **search(name,size)** in ans
29. If(ans!=-1)
30. Display the player with its data at positon **[ans]**
31. Else
32. Display player not found
33. Else if(num==4)
34. Tt=0
35. Else
36. Display enter a correct Option
37. End

**Void sort()**

1. Start
2. Declare i=0
3. While(i<5)
4. Declare max=i
5. Declare j=i
6. While(j<5)
7. If(arr[max].bowfig<=arr[j].bowfig)
8. If(arr[max].bowfig==arr[j].bowfig)
9. max=maximum of(arr[max].matches and arr[j].matches)
10. Else
11. Max=j
12. j=j+1
13. Store the value of arr[i] in temp structure.
14. Arr[i]=arr[j]
15. Arr[j]=temp
16. i=i+1
17. End.

**Void search(char name[20], int size)**

1. Start
2. Declare i=0
3. While(i<size)
4. If( arr[i].name== name)
   1. Return i
5. Else
   1. Return -1
6. i=i+1
7. End

**Void display(int size)**

1. Start
2. Print the layout.
3. Declare i=0
4. While(i<size)
   1. Display arr[i].name, arr[i].matches, arr[i].bowfig
   2. i=i+1
5. End

**Void delete(int n)**

1. Start
2. While(n<5)
   1. Arr[n-1]=arr[n]
   2. n=n+1
3. End

**Implementation details:**

#include<stdio.h>

#include<string.h>

struct players

{

char name[20];

int matchPlayed;

int bowFig;

};

struct players arr[5];

void sort()

{ //selection sort

for(int i=0;i<5;i++)

{

int max=i;

for(int j=i;j<5;j++)

{

if(arr[max].bowFig<=arr[j].bowFig){ //the bowling is given the priority

if(arr[max].bowFig==arr[j].bowFig) // if the bowling figure is same then

max=(arr[max].matchPlayed>arr[j].matchPlayed)?max:j; // compare matches played

else

max=j;

}

}

struct players temp=arr[i];

arr[i]=arr[max];

arr[max]=temp;

}

}

void display(int n){

printf("\nName\tMatches\tBowling fig.\n\n");

for(int i=0;i<n;i++)

printf("%s\t%d\t%d\n",arr[i].name,arr[i].matchPlayed,arr[i].bowFig);

}

int search(char name[20],int n)

{ //linear search

for(int i=0;i<n;i++)

if(strcmp(arr[i].name,name)==0)

return i;

return -1;

}

void delete(int n)

{

while(n++<5)

arr[n-1]=arr[n];

}

int main()

{

printf("Enter the Data\n"); //inseritng data

for(int i=0;i<5;i++)

{

printf("\n\nEnter Player name: ");

scanf("%s",arr[i].name);

printf("Enter %s's Matches played: ",arr[i].name);

scanf("%d",&arr[i].matchPlayed);

printf("Enter %s's Bowling Figure: ",arr[i].name);

scanf("%d",&arr[i].bowFig);

}

int tt=1,size=5; // using tt as break element. Size is set to 5

while(tt==1)

{

printf("\n\n1. Sort and Display\n");

printf("2. Delete a player\n");

printf("3. Search a player\n");

printf("4. Exit\n\n\n");

int num;

scanf("%d",&num);

switch(num){

case 1:

sort(); //sorting data

display(size);

break;

case 2:

printf("\n Enter which player you want to delete: ");

char s[10];

scanf("%s",s);

int n=search(s,size); //searching the postion of the element to be deleted

if(n!=-1){

delete(n); //deleting the searched element

--size;

printf("Player Deleted\n");} //decresing the size of array

else

printf("Player to be deleted not found!\n");

break;

case 3:

printf("\nEnter the player you want to search: ");

char name[20];

scanf("%s",name);

int ans=search(name,size);

if(ans!=-1)

printf("%s\t%d\t%d\n",arr[ans].name,arr[ans].matchPlayed,arr[ans].bowFig);

else

printf("Player Not Found!\n");

break;

case 4:

tt=0;

break;

default:

printf("Please Select a correct option\n\n");

}

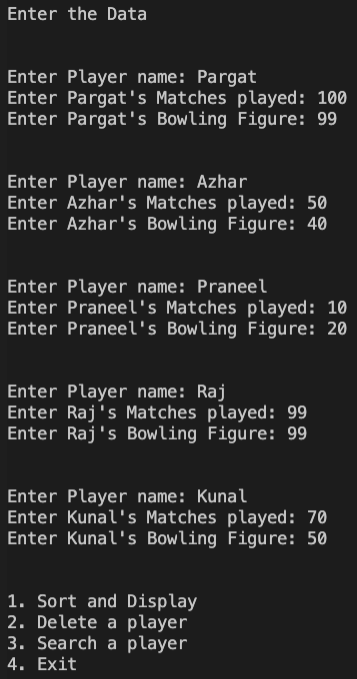
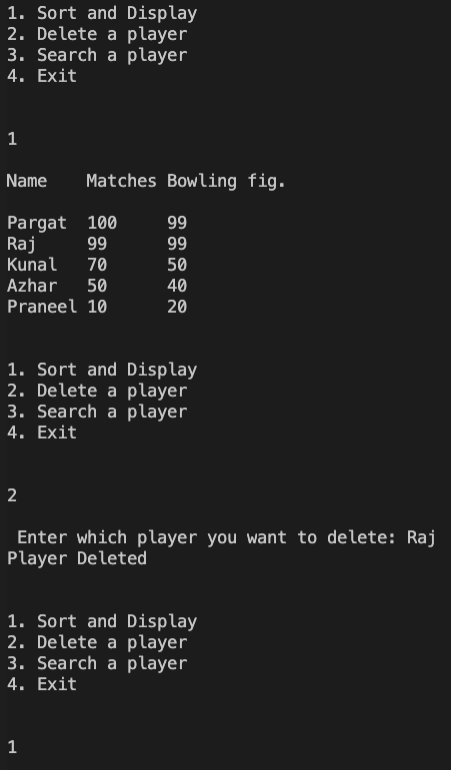
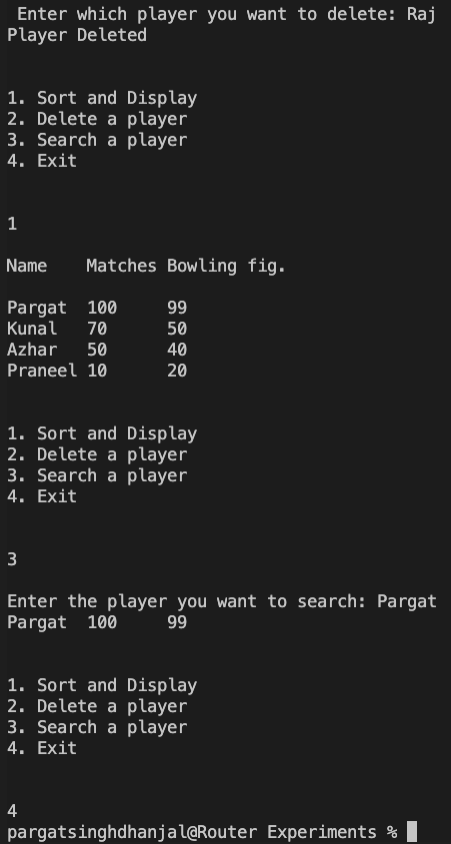
}

return 0;

}

**Output(s):**

**1 2 3**

**  **

**Conclusion:**

Successfully executed experiment 6.

Learnt to work with structure of data, sort arrays based on different conditions, delete data, search data and to manipulate structure to our benefit.

**Post Lab Descriptive Questions**

1. **Comment on the output of the following C code.**

#include <stdio.h>

struct temp

{

int a;

int b;

int c;

};

main()

{

struct temp p[] = {{1, 2, 3}, {4, 5, 6}, {7, 8, 9}};

}

**Answer: No Output.**

1. **Consider the following C code. What will be the output?**

#include<stdio.h>

struct st

{

int x;

struct st next;

};

int main()

{

struct st temp;

temp.x = 10;

temp.next = temp;

printf("%d", temp.next.x);

return 0;

}

(A) Compiler Error

(B) 10

(C) Runtime Error

(D) Garbage Value

1. **Difference between Structure and Union.**

|  |  |
| --- | --- |
| Structure | Union |
| We use **struct** keyword to declare a structure | We use **union** keyword to declare a union |
| The size of the structure is greater than or equal to the sum of sizes of its variables | The size of the union is equal to the size of the greatest variable in the union |
| Each variable of a structure has a unique memory location | Memory location is shared by the variables if the union |
| All members can be accessed at a time | Only one member can be accessed at a time |

**Date: 13/1/2022 Signature of faculty in-charge**