**Batch: B2 Roll No.: 121**

**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 data types such as arrays, strings, structures and unions.

CO4: Illustrate the use of user defined functions.

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

**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:**

Step 1: Start

Step 2: Declare Global variables i, j and num.

Step 3: Declare a structure having structure tag “players” with members name (an array of 50 characters), best\_bowl\_fig and t\_m. Declare structure variables playr (an array of 100 elements) and temp.

Step 4: Define a non-parameterized function display with void return type.

Step 5: In display function:

Step 5.1: Declare variable playr\_name (an array of 50 characters), playr\_tm, playr\_best\_bowl\_fig.

Step 5.2: Accept playr\_name, playr\_tm, playr\_best\_bowl\_fig from the user.

Step 5.3: For loop. Initialize i ß 0. Repeat the steps if i < num+1. Increment the value of i by 1 after each iteration.

Step 5.4: If i = num, copy the entered data to the respective variables in array.

Step 5.5: For loop same as that in Step 5.3. Display the data in the array in tabular form.

Step 6: Define a non-parameterized function sort with void return type.

Step 6.1: Declare variables r\_c1 and r\_c2 of integer data type.

Step 6.2: For loop. Initalize i ß 0. Repeat the steps if i < num. Increment i by 1 after each iteration.

Step 6.2.1: Nested for loop. Initialise j ß i + 1. Repeat the steps if j < num. Increment j by 1 after each iteration.

Step 6.2.2: If playr[i].best\_bowl\_fig < playr[j].bast\_bowl\_fig

Step 6.2.2.1: temp ß playr[i]

Step 6.2.2.2: playr[i] ß playr[j]

Step 6.2.2.3: playr[j] ß temp

Else if If playr[i].best\_bowl\_fig = playr[j].bast\_bowl\_fig

Step 6.2.2.4: Accept r\_c1 and r\_c2 from the user.

Step 6.2.2.5: If r\_c1 > r\_c2

Step 6.2.2.6: Repeat steps 6.2.2.1-3

Step 6.3: For loop same as that in Step 6.2. Display the sorted array.

Step 7: Define a non-parameterized function Delete with void return type.

Step 7.1: Declare a variable del\_player (an array of 50 characters).

Step 7.2: Accept del\_player from the user.

Step 7.3: For loop. Initalize i ß 0. Repeat the steps if i < num. Increment i by 1 after each iteration.

Step 7.3.1: If playr[i].name = del\_player

Continue

Step 7.3.2: Else

Print the elements of the array.

Step 8: Define a non-parameterized function search with void return type.

Step 7.1: Declare a variable search\_player (an array of 50 characters).

Step 7.2: Accept search\_player from the user.

Step 7.3: For loop. Initalize i ß 0. Repeat the steps if i < num. Increment i by 1 after each iteration.

Step 7.3.1: If playr[i].name = search\_player

Print the elements of the array (player data).

Step 7.3.2: Else

Continue.

Step 9: Define a non-parameterized function main, the main function of the program, with int return type.

Step 9.1: Declare a variable ch.

Step 9.2: Accept ch from the user.

Step 9.3: If num < 5

Display “Minimum number of players required is 5. So the program ends”

Else

Step 9.4: For loop. Initalize i ß 0. Repeat the steps if i < num. Increment i by 1 after each iteration.

Step 9.4.1: Accept playr[i].name, playr[i].t\_m and playr[i].best\_bowl\_fig

Step 9.5: Display “Enter '1' to insert data in array of structure. Enter '2' to Sort and Display. Enter '3' to Delete a player. Enter '4' to Traverse and search a player with given name.

Step 9.6: Accept ch from the user.

Case 1: Direct program control to display function.

Case 2: Direct program control to sort function.

Case 3: Direct program control to Delete function.

Case 4: Direct program control to search function.

Default: Display “The number entered is invalid”

Step 9.7: return 0

Step 10: Stop

**Implementation details:**

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

#include<math.h>

int i, j, num;

struct players

{

char name[50];

int best\_bowl\_fig; // best bowling figure

int t\_m; //total matches

}playr[100], temp;

void display()

{

char playr\_name[50];

int playr\_tm, playr\_best\_bowl\_fig;

printf("\nEnter the name of the player: ");

scanf("%s", &playr\_name);

printf("\nEnter the total number of matches\nplayed by %s: ", playr\_name);

scanf("%d", &playr\_tm);

printf("\nEnter the best bowling figure of %s: ", playr\_name);

scanf("%d", &playr\_best\_bowl\_fig);

for(i = 0; i < num+1; i++)

{

if(i = num)

{

strcpy(playr[i].name, playr\_name);

playr[i].t\_m = playr\_tm;

playr[i].best\_bowl\_fig = playr\_best\_bowl\_fig;

}

}

printf("\nThe players list is being displayed now: ");

printf("\nName\t\tTotal Matches\tBest Bowling Figure\n");

for(i = 0; i < num+1; i++)

{

printf("\n%s\t\t%d\t\t%d\n", playr[i].name, playr[i].t\_m, playr[i].best\_bowl\_fig);

}

}

void sort()

{

int r\_c1, r\_c2;

for(i = 0; i < num; i++)

{

for(j = i + 1; j < num; j++)

{

if(playr[i].best\_bowl\_fig < playr[j].best\_bowl\_fig)

{

temp = playr[i];

playr[i] = playr[j];

playr[j] = temp;

}

else if(playr[i].best\_bowl\_fig = playr[j].best\_bowl\_fig)

{

printf("\nIn this current comparison, both players have the same Best Bowling Figure.\n");

printf("\nEnter the runs conceded by %s: ", playr[i].name);

scanf("%d", &r\_c1);

printf("\nEnter the runs conceded by %s: ", playr[j].name);

scanf("%d", &r\_c2);

if(r\_c1 > r\_c2)

{

temp = playr[i];

playr[i] = playr[j];

playr[j] = temp;

}

}

}

}

printf("\nThe players list is being\ndisplayed now in the descending order: ");

printf("\nName\tTotal Matches\tBest Bowling Figure\n");

for(i = 0; i < num; i++)

{

printf("\n%s\t\t%d\t\t%d\n", playr[i].name, playr[i].t\_m, playr[i].best\_bowl\_fig);

}

}

void Delete()

{

char del\_player[50];

printf("\nEnter the name of the player you want\nto delete from the list: ");

scanf("%s", del\_player);

printf("\nThe players list with the removed\nplayer is being displayed now: ");

printf("\nName\tTotal Matches\tBest Bowling Figure\n");

for(i = 0; i < num; i++)

{

if(strcmp(playr[i].name, del\_player)==0)

continue;

else

printf("\n%s\t\t%d\t\t%d\n", playr[i].name, playr[i].t\_m, playr[i].best\_bowl\_fig);

}

}

void search()

{

char search\_player[50];

printf("\nEnter the name of the player you want\nto search from the list: ");

scanf("%s", search\_player);

printf("\nThe statistics of the searched\nplayer are being displayed below: ");

printf("\nName\tTotal Matches\tBest Bowling Figure\n");

for(i = 0; i < num; i++)

{

if(strcmp(playr[i].name, search\_player)==0)

printf("\n%s\t\t%d\t\t%d\n", playr[i].name, playr[i].t\_m, playr[i].best\_bowl\_fig);

else

continue;

}

}

int main()

{

int ch;

printf("\nEnter the number of players: ");

scanf("%d", &num);

if(num<5)

{

printf("\nMinumum number of players required is 5. So the program ends.");

exit(0);

}

else

{

for(i = 0; i < num; i++)

{

printf("\nEnter data for Record No. %d: ", i+1);

printf("\nEnter the name of the player: ");

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

printf("\nEnter the total number of matches\nplayed by %s: ", playr[i].name);

scanf("%d", &playr[i].t\_m);

printf("\nEnter the best bowling figure of %s: ", playr[i].name);

scanf("%d", &playr[i].best\_bowl\_fig);

}

}

printf("\nEnter '1' to insert data in array of structure.\nEnter '2' to Sort and Display.\nEnter '3' to Delete a player.\nEnter '4' to Traverse and search a player with given name.\n");

printf("\nEnter your choice: ");

scanf("%d", &ch);

switch(ch)

{

case 1:

display();

break;

case 2:

sort();

break;

case 3:

Delete();

break;

case 4:

search();

break;

default:

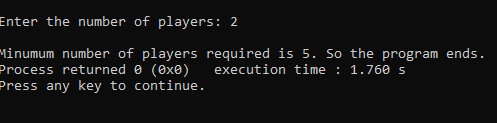
printf("\nThe number entered is invalid.");

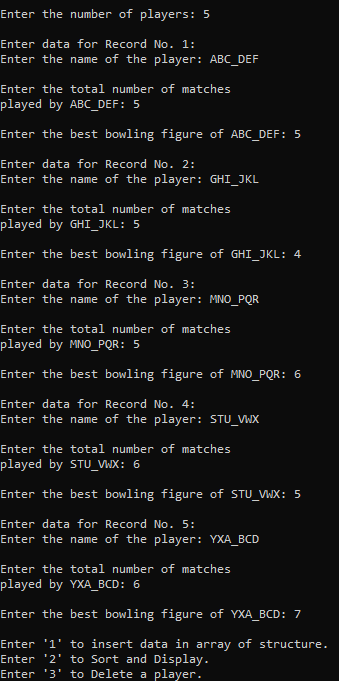
}

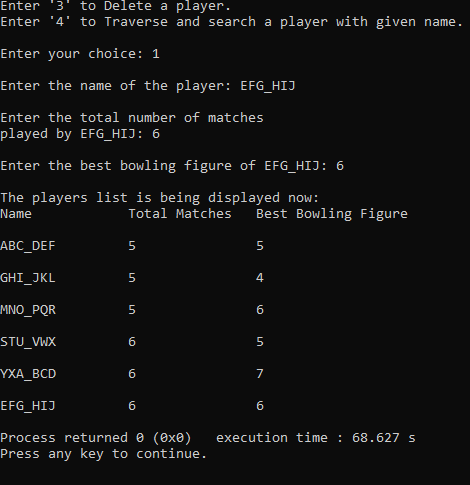
return 0;

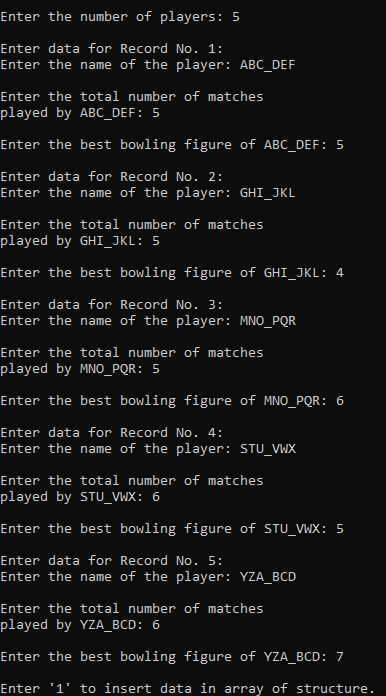
}

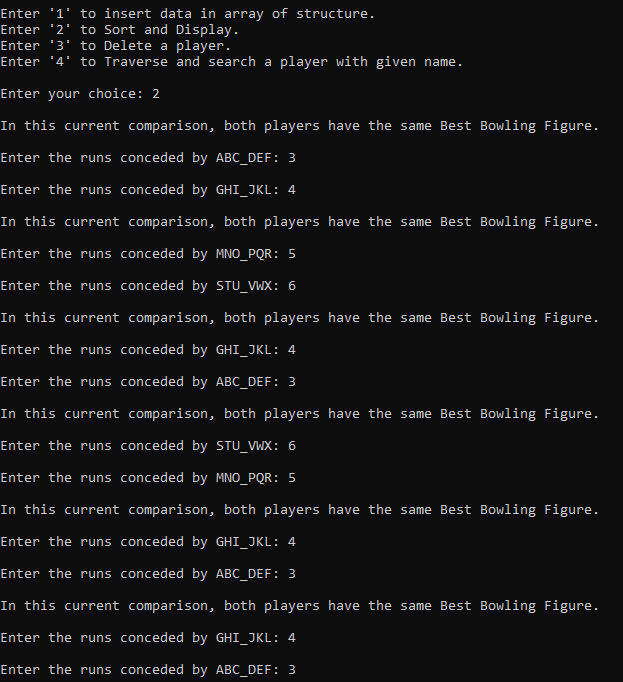
**Output(s):**

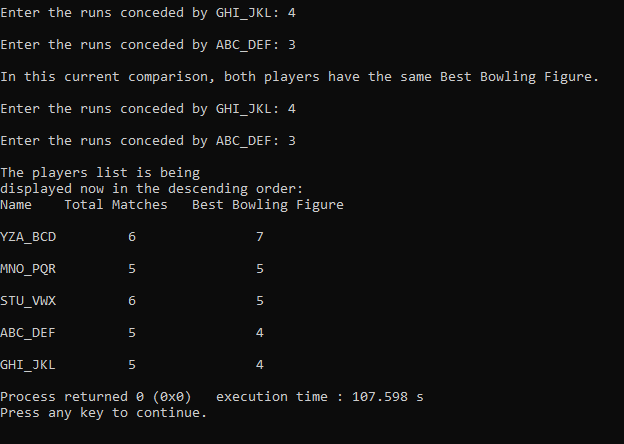


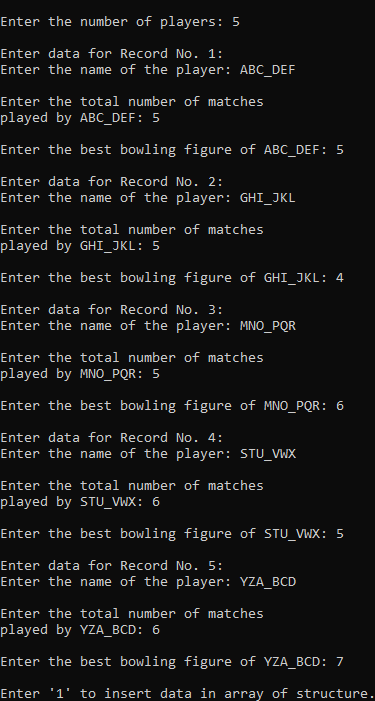


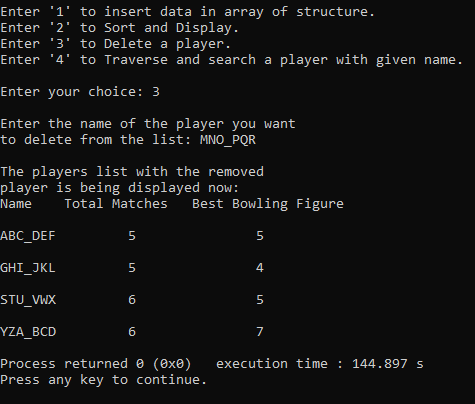


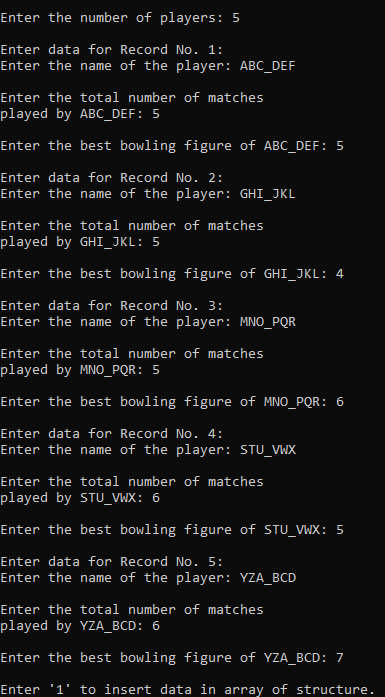


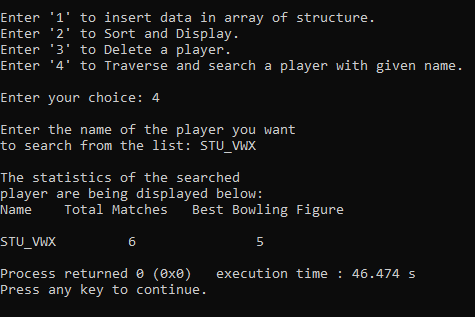












**Conclusion:**

Thus, the use of derived and structured data types such as arrays, strings, structures and unions and the use of user defined functions has been illustrated. By the combination of structures and user defined functions, the program is easier to debug and better in operation. Thus, it is easy to add more functionality to the program if needed, by adding more user defined functions. Also, the use of structures makes storing multiple data types in a single array possible, thereby enhancing the efficiency of the code.

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}};

}

Ans. The C code given above will generate no output as there is no printf() statement.

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

Ans. (A) Compiler Error

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

|  |  |  |
| --- | --- | --- |
| Sr. No. | Structure | Union |
| 1. | The keyword struct is used to define a structure. | The keyword union is used to define a union. |
| 2. | Each member of a structure has unique and independent memory location allocated to it. | All the members of the union share the same memory location whose size is equal to the size of the largest element in the union. |
| 3. | Many members of a structure can be accessed simultaneously. | Only one member of a union can be accessed at a time. |
| 4. | Many members of a structure get initialized simultaneously.. | Only the first member of a union is initialized. |
| 5. | Structure is beneficial when a large amount of data is to be stored for a long time.. | Union is not beneficial when a large amount of data must be stored for a long time. |
| 6. | Structure is not beneficial when memory space is to be conserved. | Union is beneficial when memory space is to be conserved. |

**Date: \_\_11-01-2022\_\_ Signature of faculty in-charge**