Lab 3:
Database using Array of Structure

➔ Aim

a. Create a database using array of structures and perform following operations on it. i) Add Record ii) Display Database iii) Search Record (binary search)

b. For database implemented using array, perform iv) Modify record v) Delete record vi) Sort Database (Bubble sort)

➔ Theory

● What is Array of Structure?

An array of structures can be defined as the collection of multiple structures variables where each variable contains information about different entities. The array of structures are used to store information about multiple entities of different data types. The array of structures is also known as the collection of structures. Following diagram shows how array of structure works.

(Fig: Array of Structure)

● Pseudocode for Creating the database using Array of Structure:

```
/* s[100] can create entries of 100 students with their roll no
Name and address*/
struct student
{
unsigned int roll;
char name[20];
char addr[20];
}s[100],temp;
/*This code accepts the n number of entries and creates the
Database */
void getData(int n)
{
int i;
cout<<"\nEnter the records:\n";
for(i=0;i<n;i++)
{
cin>>s[i].roll>>s[i].name>>s[i].addr;
}
}
```

● Pseudocode for Displaying the database

```
void displayData(int n)
{
int i;
cout<<"\nRollNo\tName\tAddress\n";
cout<<"-------------------------";
for(i=0;i<n;i++)
{
cout<<s[i].roll<<"\t"<<s[i].name<<"\t"<<s[i].addr;
}
cout<<"-------------------------";
}
```

● Pseudocode for Adding Record into Database

```
int insertRecord(int n)
{
int i,pos;
cout<<"\nEnter the position of record";
cin>>pos;
for(i=n-1;i>=pos-1;i--)//(1,1,--)
{
s[i+1]=s[i];
}
cout<<"\nRoll no:\t";
cin>>s[pos-1].roll;
cout<<"\nName:\t";
cin>>s[pos-1].name;
cout<<"\nAddress:\t";
cin>>s[pos-1].addr;
return n+1; //reset array size
}
```

● Pseudocode for Deleting the record from the database

```
int deleteRecord(int n)
{
int pos,i;
cout<<"\nEnter the position of the element to be deleted";
cin>>pos;
for(i=pos-1;i<n-1;i++)
{
s[i]=s[i+1];
}
n=n-1;
return n;
}
```

● **Pseudocode to Modify a record.**

```
void updateRecord()
{
int rno,i=0;
cout<<"\nEnter the roll number to update the record";
cin>>rno;
while(s[i].roll!=rno)
{
i++;
}
cout<<"\nEnter the new record";
cin>>s[i].roll>>s[i].name>>s[i].addr;
}
```

● **Bubble Sort.**

It is implemented by shifting the largest number to last position then decrement the last and so on.

Example: 14 33 27 35 10

Pass 1: **14 33** 27 35 10

14 **33 27** 35 10

14 27 **33 35** 10

14 27 33 **35 10**

Pass 2: **14 27** 33 10 35

14 **27 33** 10 35

14 27 **33 10** 35

Pass 3: **14 27** 10 33 35

14 **27 10** 33 35

Pass 4: **14 10** 27 33 35

Sorted: 10 14 27 33 35

Complexity: $O(n^2)$

● **Pseudocode**

```
void sortData(int n)
{
int i,j;
for(i=0;i<n-1;i++)
{
for(j=0;j<n-i-1;j++)
{
if(s[j].roll>s[j+1].roll)
{
temp=s[j];
s[j]=s[j+1];
s[j+1]=temp;
}
}
```

}
}

● **Binary Search**

For searching the data using binary search the database should be sorted.
It works by dividing the list into two subparts until the middle element and search element matches. Complexity is $O(log_2 n)$.
Example: If we want to search 31 in the given list

0 1 2 3 4 5 6 7 8
10 14 19 26 **27** **31** 33 35 42
Begin = 0,
End = 8,
Mid = (begin+end)/2 = (0+8)/2 = 4
As element 27 at position 4 is less than 31 therefore,
Begin = mid+1 = 4+1= 5
End = 8
Mid = (begin+end)/2 = (5+8)/2=7
5 6 7 8
31 33 **35** 42
As element 35 at position 7 is greater than 31 therefore
Begin =5
End = mid-1= 7-1= 6
Mid = (begin+end)/2 =(5+6)/2=6
5 6 7
31 **33** 35
As element 33 at position 6 is greater than 31 therefore
Begin = 5
End =mid-1= 6-1=5
Mid =(begin+end)/2 = (5+5)/2=5
5
31

Here the middle element matches with the search element and the position of search element is 5. Therefore 31 is found at location 5.
If the value of the end is smaller than begin we can say that the element is not present in the list. Therefore the comparisons will be done only if the value of begin is smaller than or equal to the value of end.
Binary search requires repetitive dividing of the list, therefore it can be performed by using loops or by using recursion by calling the function itself until the middle element matches with the search element.
The pseudocode given below does not uses recursion. The searching is performed using while loop.
Complexity: $O(log_2 n)$

● **Pseudocode**

```
void searchData(int n)
{
int begin,end,mid,i,j,rno;
cout<<"\nEnter the roll number to search the record: ";
```

```
cin>>rno;
begin=0;
end=n-1;
mid=(begin+end)/2;
while(begin<=end)
{
if(s[mid].roll<rno)
{
begin=mid+1;
}
else if(s[mid].roll==rno)
{
cout<<”Record Found”;
cout<<s[mid].roll<<s[mid].name<<s[mid].addr;
Break;
}
else
{
end=mid-1;
}
mid=(begin+end)/2;
}
if(begin>end)
{
cout<<"Record not found";
}
}
```

Program:

```
#include<iostream.h>
#include<iomanip.h>
#include<stdlib.h>
struct stud{
int rollNo;
char nm[20];
char addr[20];
}s[100];
void getdata(int n)
{

for(int i=0;i<n;i++)
{
cout<<endl;
cout<<"Enter roll no,name and address";
cin>>s[i].rollNo>>s[i].nm>>s[i].addr;
}
}
void display(int n)
{
for(int i=0;i<n;i++)
{
cout<<endl;
cout<<"roll no:"<<s[i].rollNo<<endl<<"name:"<<s[i].nm<<endl<<"address:"<<s[i].addr<<endl;
}
}
```

```
void bubble_sort(stud list[],int n)
{
stud s1;
int iteration;
int index;
for(iteration=1;iteration<n;iteration++)
{
for(index=0;index<n-iteration;index++)
if(list[index].rollNo>list[index+1].rollNo)
{
s1=list[index];
list[index]=list[index+1];
list[index+1]=s1;
}
}
}
void binary(stud list[],int key,int lb,int ub)
{
int mid;
while(lb<=ub)
{
mid=(lb+ub)/2;
if(key==list[mid].rollNo)
{
cout<<"element found at location "<<mid+1<<endl;
break;
}
else
```

```
{
if(key>list[mid].rollNo)
lb=mid+1;
else
ub=mid-1;
}
}
}
void modify(stud list[],int n)
{
int tp;
cout<<"please enter the roll no of student you would like to edit";
cin>>tp;
for(int i=0;i<n;i++)
{
if(tp==list[i].rollNo)
{
cout<<endl;

cout<<"Enter updated roll no,name and address of student"<<endl;
cin>>s[i].rollNo>>s[i].nm>>s[i].addr;
}
else
{
cout<<"Please enter valid roll no"<<endl;
}
}
}
```

```
int main()
{

int n,choice;
int key;

do{
cout<<"Enter ur choice:"<<endl<<"1.getdata"<<endl<<"2.display"<<endl<<"3.bubble
sort"<<endl<<"4.binary search"<<endl<<"5.modify"<<endl<<"6.Exit"<<endl;
cin>>choice;
switch(choice)
{

case 1:
cout<<"enter how many records:";
cin>>n;
getdata(n);
break;
case 2:
if(n<0)
{
cout<<"accept data first";
break;
}
else{
display(n);
break;
```

```
}
case 3:
bubble_sort(s,n);
cout<<"array of structure is sorted";
display(n);
break;
case 4:
cout<<"Enter element to be search:";
cin>>key;
binary(s,key,0,n-1);
break;
case 5:
modify(s,n);
break;

}
}while(choice!=6);
system("pause");

return 0;
}
```

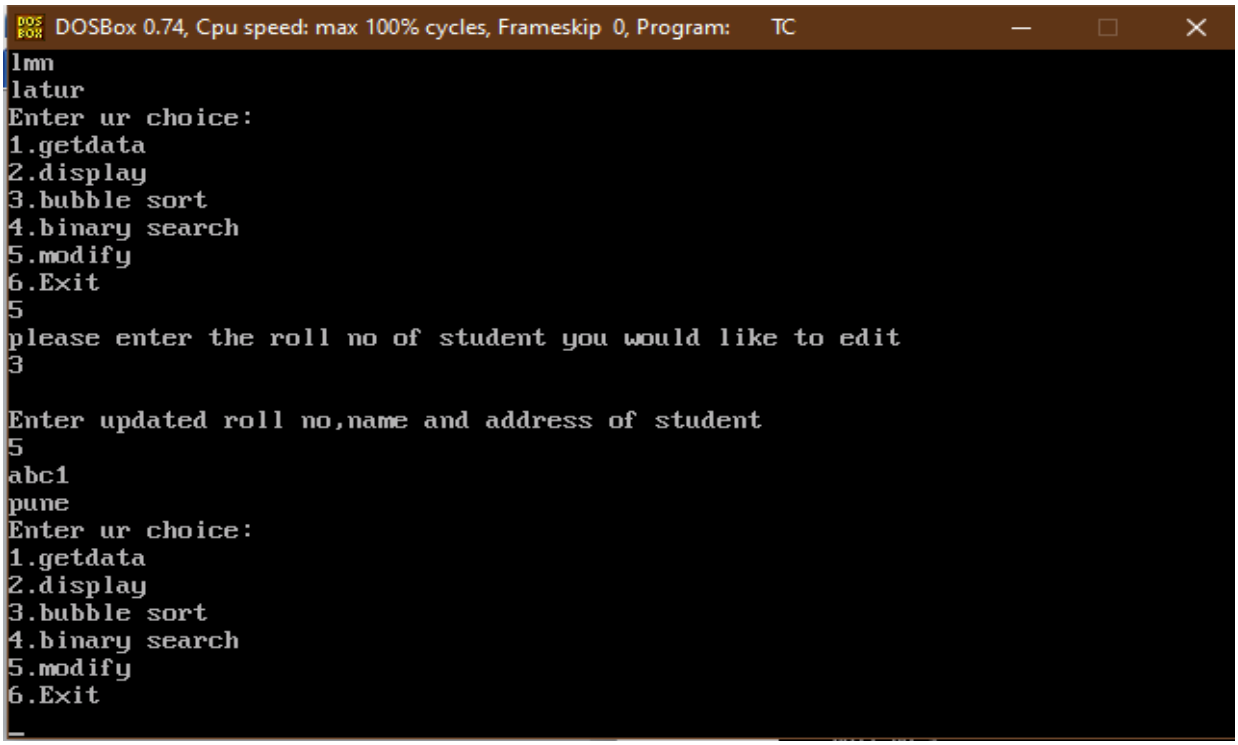
Output:

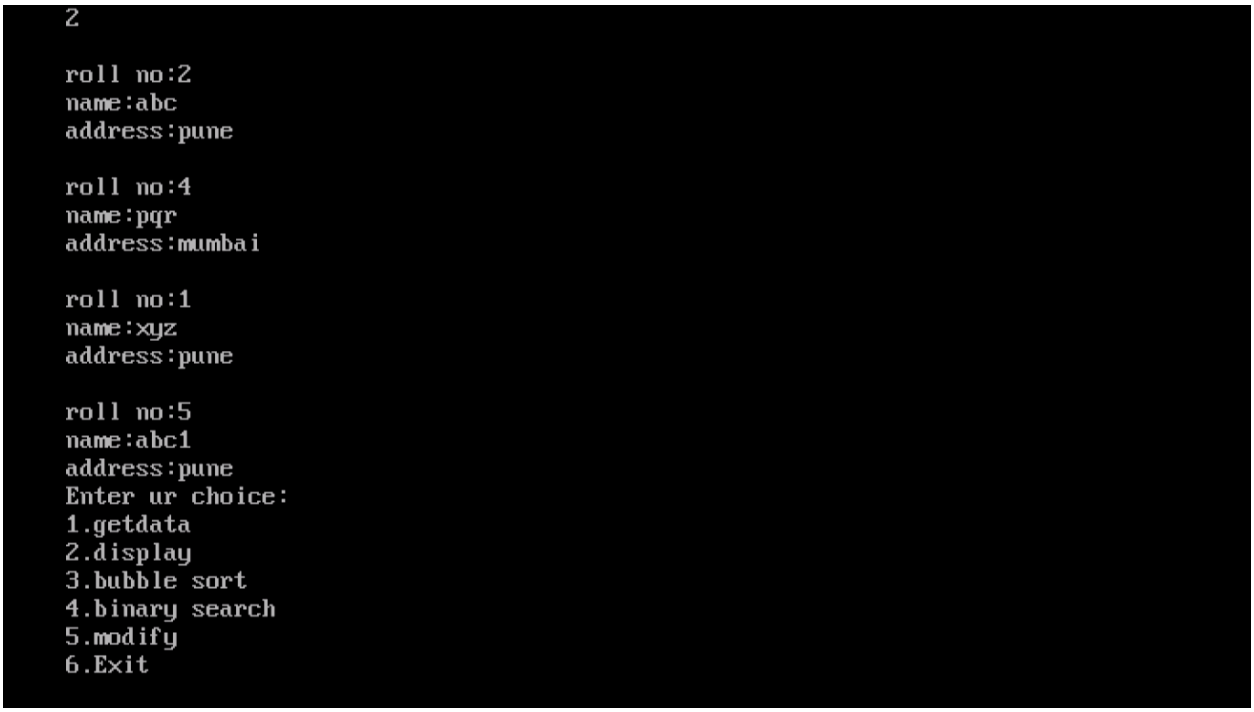
```
3.bubble sort
4.binary search
5.modify
6.Exit
1
enter how many records:4

Enter roll no,name and address2
abc
pune

Enter roll no,name and address
4
pqr
mumbai

Enter roll no,name and address
1
xyz
pune

Enter roll no,name and address
3
lmn
latur
```

```
2

roll no:2
name:abc
address:pune

roll no:4
name:pqr
address:mumbai

roll no:1
name:xyz
address:pune

roll no:3
name:lmn
address:latur
Enter ur choice:
1.getdata
2.display
3.bubble sort
4.binary search
5.modify
6.Exit
```

```
3
array of structure is sorted
roll no:1
name:xyz
address:pune

roll no:2
name:abc
address:pune

roll no:3
name:lmn
address:latur

roll no:4
name:pqr
address:mumbai
Enter ur choice:
1.getdata
2.display
3.bubble sort
4.binary search
5.modify
6.Exit
_
```

```
roll no:3
name:lmn
address:latur

roll no:4
name:pqr
address:mumbai
Enter ur choice:
1.getdata
2.display
3.bubble sort
4.binary search
5.modify
6.Exit
4
Enter element to be search:3
element found at location 3
Enter ur choice:
1.getdata
2.display
3.bubble sort
4.binary search
5.modify
6.Exit
_
```


DOSBox 0.74, Cpu speed: max 100% cycles, Frameskip 0, Program: TC
lmn
latur
Enter ur choice:
1.getdata
2.display
3.bubble sort
4.binary search
5.modify
6.Exit
5
please enter the roll no of student you would like to edit
3
Enter updated roll no,name and address of student
5
abc1
pune
Enter ur choice:
1.getdata
2.display
3.bubble sort
4.binary search
5.modify
6.Exit


2
roll no:2
name:abc
address:pune
roll no:4
name:pqr
address:mumbai
roll no:1
name:xyz
address:pune
roll no:5
name:abc1
address:pune
Enter ur choice:
1.getdata
2.display
3.bubble sort
4.binary search
5.modify
6.Exit

➔ **Conclusion**

In this way we have successfully studied how to create an array of structures and perform various operations on it such as Add, Delete, Modify, Display, Sort and Search.