Practical 1 :- Write a program to demonstrate single dimensional array.

```
#include <iostream.h>
#include <conio.h>
int main() {
   clrscr();
   int arr[5] = {10, 20, 30, 40, 50};
   cout << "Single Dimensional Array Elements:\n";
   for (int i = 0; i < 5; i++) {
      cout << arr[i] << " ";
   }
   getch();
   return 0;
}
```

Output:

```
Single Dimensional Array Elements:
10 20 30 40 50
```

Practical 2 :- Write a program to demonstrate multidimensional array.

```
#include <iostream.h>
#include <conio.h>
int main() {
   clrscr();
   int arr[2][3] = {{1,2,3},{4,5,6}};
   cout << "Two Dimensional Array Elements:\n";
   for (int i=0;i<2;i++) {
      for (int j=0;j<3;j++) {
         cout << arr[i][j] << " ";
      }
      cout << "\n";
   }
getch();
 return 0;
}
```

Output:

```
Two Dimensional Array Elements:
1 2 3
4 5 6
```

Practical 3 :- Write a program to implement Bubble Sort.

```
#include <iostream.h>
#include <conio.h>
int main() {
   clrscr();
   int n, i, j, temp;
   cout << "Enter size of array: ";
   cin >> n;
   int arr[50];
   cout << "Enter elements:\n";
   for(i=0;i<n;i++) cin >> arr[i];
   for(i=0;i<n-1;i++) {
      for(j=0;j<n-i-1;j++) {
         if(arr[j] > arr[j+1]) {
            temp = arr[j];
            arr[j] = arr[j+1];
            arr[j+1] = temp;
         }
      }
   }
   cout << "Sorted Array:\n";
   for(i=0;i<n;i++) cout << arr[i] << " ";
   getch();
   return 0;
}
```

Input: 5 elements → 64 25 12 22 11

Output: Sorted Array: 11 12 22 25 64

Practical 4 :- Write a program to implement Quick Sort.

```
#include <iostream.h>
#include <conio.h>
void quicksort(int a[], int first, int last) {
    int i, j, pivot, temp;
    if(first < last) {
        pivot = first;
        i = first;
        j = last;
        while(i < j) {
            while(a[i] <= a[pivot] && i < last) i++;
            while(a[j] > a[pivot]) j--;
            if(i < j) {
                temp = a[i];
                a[i] = a[j];
                a[j] = temp;
            }
        }
        temp = a[pivot];
        a[pivot] = a[j];
        a[j] = temp;
        quicksort(a, first, j-1);
        quicksort(a, j+1, last);
    }
}
int main() {
    clrscr();
    int n, i;
    cout << "Enter number of elements: ";
    cin >> n;
    int arr[50];
    cout << "Enter elements:\n";
    for(i=0;i<n;i++) cin >> arr[i];
    quicksort(arr,0,n-1);
    cout << "Sorted array:\n";
    for(i=0;i<n;i++) cout << arr[i] << " ";
    getch();
    return 0;
}
```

Input: 6 elements → 34 7 23 32 5 62

Output:

```
Sorted array: 5 7 23 32 34 62
```

Practical 5 :- Write a program to implement Selection Sort.

```
#include <iostream.h>
#include <conio.h>
int main() {
   clrscr();
   int n, i, j, min, temp;
   cout << "Enter size of array: ";
   cin >> n;
   int arr[50];
   cout << "Enter elements:\n";
   for(i=0;i<n;i++) cin >> arr[i];
   for(i=0;i<n-1;i++) {
      min = i;
      for(j=i+1;j<n;j++) {
         if(arr[j] < arr[min])
            min = j;
      }
      temp = arr[i];
      arr[i] = arr[min];
      arr[min] = temp;
   }
   cout << "Sorted Array:\n";
   for(i=0;i<n;i++) cout << arr[i] << " ";
   getch();
   return 0;
}
```

Input: 5 elements → 29 10 14 37 13 **Output:** Sorted Array: 10 13 14 29 37

Practical 6 :- Write a program to implement Insertion Sort.

```
#include <iostream.h>
#include <conio.h>
int main() {
    clrscr();
    int n, i, j, key;
    cout << "Enter size of array: ";
    cin >> n;
    int arr[50];
    cout << "Enter elements:\n";
    for(i=0;i<n;i++) cin >> arr[i];
    for(i=1;i<n;i++) {
        key = arr[i];
        j = i-1;
        while(j>=0 && arr[j] > key) {
            arr[j+1] = arr[j];
            j--;
        }
        arr[j+1] = key;
    }
    cout << "Sorted Array:\n";
    for(i=0;i<n;i++) cout << arr[i] << " ";
    getch();
    return 0;
}
```

Input: 6 elements → 12 11 13 5 6 7
Output: Sorted Array: 5 6 7 11 12 13

Practical 7 :- Write a program to implement Linear Search.

```
#include <iostream.h>
#include <conio.h>
int main() {
   clrscr();
   int n, i, key, flag=0;
   cout << "Enter size of array: ";
   cin >> n;
   int arr[50];
   cout << "Enter elements:\n";
   for(i=0;i<n;i++) cin >> arr[i];
   cout << "Enter element to search: ";
   cin >> key;
   for(i=0;i<n;i++) {
      if(arr[i]==key) {
         cout << "Element found at position " << i+1;
         flag=1;
         break;
      }
   }
   if(flag==0) cout << "Element not found";
   getch();
   return 0;
}
```

Input: 5 elements → 10 20 30 40 50, Search = 30

Output:

```
Element found at position 3        (If searched for 99) → Element not found
```

Practical 8 :- Write a program to implement Binary Search.

```
#include <iostream.h>
#include <conio.h>
int main() {
   clrscr();
   int n, i, key, low, high, mid, flag=0;
   cout << "Enter size of array: ";
   cin >> n;
   int arr[50];
   cout << "Enter elements in sorted order:\n";
   for(i=0;i<n;i++) cin >> arr[i];

   cout << "Enter element to search: ";
   cin >> key;

   low=0; high=n-1;
   while(low<=high) {
      mid=(low+high)/2;
      if(arr[mid]==key) {
         cout << "Element found at position " << mid+1;
         flag=1; break;
      }
      else if(arr[mid]<key) low=mid+1;
      else high=mid-1;
   }
   if(flag==0) cout << "Element not found";
   getch();
   return 0;
}
```

Input: Sorted Array = 10 20 30 40 50, Search = 40
Output:

Element found at position 4

(If searched for 15) → Element not found

Practical 9 :- Write a program to implement Stack Operations :- PUSH, POP, DISPLAY.

```
#include <iostream.h>
#include <conio.h>
#define MAX 10
int stack[MAX], top=-1;
void push(int x) {
  if(top==MAX-1) cout<<"Stack Overflow\n";
  else stack[++top]=x;
}
void pop() {
  if(top==-1) cout<<"Stack Underflow\n";
  else cout<<"Popped: "<<stack[top--]<<"\n";
}
void display() {
  if(top==-1) cout<<"Stack Empty\n";
  else {
    cout<<"Stack elements:\n";
    for(int i=top;i>=0;i--) cout<<stack[i]<<" ";
    cout<<"\n";
  }
}
int main() {
  clrscr();
  int ch, val;
  do {
    cout<<"\n1.Push 2.Pop 3.Display 4.Exit\n";
    cin>>ch;
```

```
        switch(ch) {
            case 1: cout<<"Enter value: "; cin>>val; push(val); break;
            case 2: pop(); break;
            case 3: display(); break;
        }
    } while(ch!=4);
    getch();
    return 0;
}
```

Input/Choice Example:

```
1.Push
2.Pop
3.Display
4.Exit

Choice: 1
Enter value: 10
Choice: 1
Enter value: 20
Choice: 3
```

Output:

```
Stack elements: 20 10
After Pop → Popped: 20
```

Practical 10 :- Write a program to implement Linear Queue Operations :- INSERT, DELETE, DISPLAY.

```
#include <iostream.h>
#include <conio.h>
#define MAX 10
int q[MAX], front=-1, rear=-1;
void insert(int x) {
   if(rear==MAX-1) cout<<"Queue Overflow\n";
   else {
      if(front==-1) front=0;
      q[++rear]=x;
   }
}
void del() {
   if(front==-1 || front>rear) cout<<"Queue Underflow\n";
   else cout<<"Deleted: "<<q[front++]<<"\n";
}
void display() {
   if(front==-1 || front>rear) cout<<"Queue Empty\n";
   else {
      cout<<"Queue elements:\n";
      for(int i=front;i<=rear;i++) cout<<q[i]<<" ";
      cout<<"\n";
   }
}
int main() {
   clrscr();
   int ch, val;
```

```
  do {
     cout<<"\n1.Insert 2.Delete 3.Display 4.Exit\n";
     cin>>ch;
     switch(ch) {
        case 1: cout<<"Enter value: "; cin>>val; insert(val); break;
        case 2: del(); break;
        case 3: display(); break;
     }
  }
while(ch!=4);
  getch();
  return 0;
}
```

Input/Choice Example:

```
1.Insert
2.Delete
3.Display
4.Exit

Choice: 1
Enter value: 5
Choice: 1
Enter value: 10
Choice: 3
```

Output:

```
Queue elements: 5 10
After Delete → Deleted: 5
```

Practical 11 :- Write a program to implement Singly Linked list with operations :- CREATE, INSERT, DELETE.

```
#include <iostream.h>
#include <conio.h>
#include <stdlib.h>
struct Node {
   int data;
   Node *next;
};
Node *head=NULL;
void create(int val) {
   Node *newNode=new Node;
   newNode->data=val;
   newNode->next=head;
   head=newNode;
}
void insert(int val) {
   Node *newNode=new Node;
   newNode->data=val;
   newNode->next=NULL;
   if(head==NULL) head=newNode;
   else {
      Node *temp=head;
      while(temp->next!=NULL) temp=temp->next;
      temp->next=newNode;
   }
}
void del(int val) {
    Node *temp=head, *prev=NULL;
    while(temp!=NULL && temp->data!=val) {
        prev=temp; temp=temp->next;
    }
    if(temp==NULL) cout<<"Not Found\n";
    else {
        if(prev==NULL) head=temp->next;
        else prev->next=temp->next;
        delete temp;
    }
}
void display() {
    Node *temp=head;
    while(temp!=NULL) {
        cout<<temp->data<<" -> ";
        temp=temp->next;
    }
    cout<<"NULL\n";
}
int main() {
    clrscr();
    int ch,val;
    do {
        cout<<"\n1.Create 2.Insert 3.Delete 4.Display 5.Exit\n";
        cin>>ch;
        switch(ch) {
            case 1: cout<<"Enter value: "; cin>>val; create(val); break;
            case 2: cout<<"Enter value: "; cin>>val; insert(val); break;
            case 3: cout<<"Enter value: "; cin>>val; del(val); break;
            case 4: display(); break;
        }
    } while(ch!=5);
    getch();
    return 0;
}
```

Input Example:

- Create(10), Insert(20), Insert(30), Display

Output:

```
10 -> 20 -> 30 -> NULL
After Delete(20): 10 -> 30 -> NULL
```

Practical 12 :- Write a program to implement Doubly Linked List.

```
#include <iostream.h>
#include <conio.h>
#include <stdlib.h>
struct Node {
   int data;
   Node *prev, *next;
};
Node *head=NULL;
void insert(int val) {
   Node *newNode=new Node;
   newNode->data=val;
   newNode->prev=NULL;
   newNode->next=head;
   if(head!=NULL) head->prev=newNode;
   head=newNode;
}
void del(int val) {
   Node *temp=head;
   while(temp!=NULL && temp->data!=val) temp=temp->next;
   if(temp==NULL) cout<<"Not Found\n";
   else {
      if(temp->prev!=NULL) temp->prev->next=temp->next;
      else head=temp->next;
      if(temp->next!=NULL) temp->next->prev=temp->prev;
      delete temp;
   }
}
```

```
void display() {
    Node *temp=head;
    while(temp!=NULL) {
        cout<<temp->data<<" <-> ";
        temp=temp->next;
    }
    cout<<"NULL\n";
}
int main() {
    clrscr();
    int ch,val;
    do {
        cout<<"\n1.Insert 2.Delete 3.Display 4.Exit\n";
        cin>>ch;
        switch(ch) {
            case 1: cout<<"Enter value: "; cin>>val; insert(val); break;
            case 2: cout<<"Enter value: "; cin>>val; del(val); break;
            case 3: display(); break;
        }
    } while(ch!=4);
    getch();
    return 0;
}
```

Input Example:

- Insert(10), Insert(20), Insert(30), Display

Output:

```
30 <-> 20 <-> 10 <-> NULL
After Delete(20): 30 <-> 10 <-> NULL
```