S.No	Names of the programs	page no	Signature
1.	Sorted Array (Bubble Sort)		
2.	Merge Sort		
3.	Binary Search		
4.	Linear Search		
5.	Depth first Search		
6.	Stack using pointer		
7.	Stack using Array		
8.	Queue using pointer		
9.	Queue using Array		
10.	Inorder Traversal		
11.	Preorder Traversal		
12.	Post order Traversal		
13.	Insertion Sort		

1. Bubble Sort

Aim:

To Sort the given array of values using c ++ program.

program Coding:

```
#include<iostream.h>
#include<conio.h>
void main()
{
   clrscr();
   int a[10], i, j, n, t;
   cout<<"Enter total number of element:";
   cin>>n;
   cout<<"\n Enter array element:";
   for(i=1;i<=n;i++)
   cin>>a[i];
   for(i=1;i<=n;i++)
   for(j=i+1;j<=n;j++)
```

Output:

```
Enter total number of element : 3

Enter array element:

90
85
65

Sorted data:

65
85
90
```

```
if (a[i] > a[j])
{
    t = a[i];
    a[i] = a[j];
    a[j] = t;
}
cout << "\n Sorted data";
for (i=1; i<=n; i++)
cout << a[i] << "\n";
getch ();
}
```

Result:

Thus the given array of value is sorted.

2. Merge Sort

Aim:

To array the given number using Merge Sort.

Program Coding:

```
# Include <iostream.h>
# Include <conio.h>
void ms (int, int);
void merge (int, int, int);
int a[10], n, i, j;
void main()
{
    clrscr();
    Cout << " Enter Number of Element:";
    Cin >> n;
    Cout << " \n Enter array elements:";
    for (i=1; i<n; i++)
    Cin >> a[i];
    ms(1,n);
```

```
cout << "\n Sorting order:";
for (i=1;i<=n;i++)
cout << "\n" << a[i];
getch();
}
void ms (int low, int high)
{
if (low < high)
{
int mid = (low +high) / 2 ;
ms (low, mid);
ms (mid +1, high);
merge (low, mid, high);
}
}
void merge (int low, int mid, int high)
{
int h= low, i= mid +1, k= low, b [10];
while (h <= mid && i <= high)
if (a [h] <= a [i])
```

```
        b [k++] = a [h++];
    else
        b [k++] = a [i++];
    if (h > low)
        for (int j = i; j <= high; j++)
        b [k++] = a [j];
    else if (i > h)
        for (int j=h; j <= mid ; j++)
        b [k++] = a [j];
        for (int j = low; j <= high ; j++)
         a [i] = b [j];
}
```

Output:

```
Enter number of Element :
5
Enter array element:
20
40
500
700
30
Sorted order:
20
30
40
500
700
```

Result:

Thus the given numbers are Sorted using merge Sort.

3- Bfhary Search

Aim :

To Search the given number using binary Search method.

program Coding:

```
#Include < &Postream.h>
#Include <& Conio.h>
Void main ()
{
 Pnt a [100], n, i, beg, end, mid, data;
  ClrScr ();
  Cout << " Enter number of element-";
  Cin >> n;
  Cout << " In Enter array element In Sorted
                                order :";
  for (i=1; i<=n ; i++)
  Cin >> a [i];
  Cout << " In Enter data want to Search-";
  cin >> data;
```

```
beg = 1;
end = n;
mid = n/2;
while ((beg < end) && (a[mid] != data))
{
if (a[mid] > data)
{
   end = mid - 1;
}
else
 {
  beg = mid + 1;
 }
  mid = (beg + end)/2;
}
if (a[mid] == data)
  cout << " In Data is found";
else
  cout << " In Data is not found";
  getch ();
}
```

Output:

Enter number of element = 4

Enter array element in Sorted order:

7
8
9
10

Enter data want to Search = 7

Data is found

Result:

Thus the given number is Searched using binary Search method.

4. Linear Search

Aim:

To Search the given number using Linear Search method.

program Coding:-

```
#Include< Io Stream.h>
#Include< Conio.h>
#Include< process.h>
Void main ()
{
Int a [10], se, i, n;
clrscr ();
Cout <<" Enter number of element -";
cin >>n;
Cout <<" In Enter array element :";
for (i=1;i <n; i++)
cin >> a [i];
cin >> a [i][j];
```

Output:

Enter number of element -3

Enter array element :

79

30

100

Enter Search element :

30

Element is found.

```
cout << "\n Enter Search element :";
cin >> se;
for (i=1; i<=n; i++)
{
  if(a [i] == se)
  {
   cout << "\n "Element is found.";
    getch ();
    exit (0);
  }
}
  cout << "\n Element is not found.";
   getch ();
}
```

Result:

Thus the given number is Searched using linear Search method.

5. Depth first Search

Aim:

To Search the given number using depth first Search method.

Program Coding:

```
#Include <iostream.h>
#Include <conio.h>
class graph
{
 Int a [10] [10];
 Int V[10];
 Int n;
 Public:
  Void getgraph ()
  {
   Cout << " Enter number of nodes - ";
   cin >> n;
   for (int i = 1; i <= n; i++)
```

```
for (int j=1; j<n; j++)
cin >> a[i][j];
for (i=1; i<n; i++)
  v[i]=0;
  dfs(i);
}
void dfs(int x)
{
 cout<<" \n visit:"<<x<<"\n";
  v[x]=1;
  for(int i=1; i<=n; i++)
  if (a[x][i]==1)
  if (!v[i])
  {
    dfs(i);
  }
 }
};
void main()
{
  clrscr();
```

Depth First Search

Output :-

Enter the number of nodes - 6

0 1 1 1 0 0

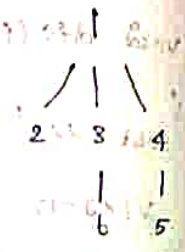
1 0 0 0 0 0

1 0 0 0 0 1

1 0 0 0 1 0

0 0 0 1 0 0

0 0 1 0 0 0



Visit : 1

Visit : 2

Visit : 3

visit : 6

Visit : 4

Visit : 5

```
graph g;
g.getGraph();
getch();
}
```

Result:

Thus the given number is Searched using depth first search method.

6. Stack using pointer

Aim:

To implement a stack as a linked list using pointer.

Program Coding:

```
#include<iostream.h>
#include<conio.h>
#include<stdlib.h>
Class node
{
  int data;
  node * next;
  Public;
     node ()
   {
    data =0; next =0;
   }
   friend Class Stack;
};
```

```
class stack
{
 node* top;
 public:
   Stack ()
    {
     top = 0;
    }
    void push (int x);
    void pop ();
     void display ();
};
void stack :: push (int x)
 {
  node * temp = new node;
  temp -> data = x;
  if (top == 0)
  top = temp;
  else {
     temp -> next = top;
     top = temp; }
  }
```

```
void stack :: pop ()
{
if (top ==0)
  cout << "\n Stack empty.";
  else
  {
  cout << "\n popped =" << top ->data;
  top = top ->next;
  }
}
void stack :: display ()
{
for (node* temp =top; temp = 0; temp = temp->next)
cout <<temp -> data << "\n";
}
void main ()
{
  int item c;
    clrscr ();
    Stack s;
    do
    {
```

```
Cout << "\n 1. push  2. pop  3. display  4. exit.-";
cin >> c;
Switch (c)
{
 Case 1:
    Cout << "\n Enter Item =";
     cin >> Item;
     s. push (Item);
      break;
   Case 2:
       s. pop ();
       break;
   case 3:
       s. display ();
       break;
     default:
        Exit (0);
}
getch ();
}
while (1);
}
```

Output:

1. push 2. pop 3. display 4. exit - 1

Enter Item = 100

1. push 2. pop 3. display 4. exit - 1

Enter Item = 200

1. push 2. pop 3. display 4. exit - 2

Poffed = 200

1. push 2. pop 3. display 4. exit - 3

100

Result:

Thus the Stack is implemented as a linked list using pointer.

7. Stack using Array

Aim:

To Implement a stack as a linked list using to create, push, pop a stack using Array.

Program Coding:

```
#include <iostream.h>
#include <conio.h>
#include<stdlib.h>
int top=0,a[10],i,n, data;
class stack
 {
  public:
    void push()
     {
      if (top >=5)
      {
      cout <<" In Stack is full";
      }
     else
     {
```

```
    cout << "\n  Enter data ;
    cin >> data;
    a [top] = data;
    top ++;
  }
}
void pop ()
 {
  if (top <= 0)
    cout << "\n Stack is empty.";
  else
    -- top;
    cout << a [top];
  }
void display ()
 {
  if (top == 0)
   cout << "\n No data in the stack.";
  else
  {
   for (i=top -1; i >= 0; i --)
```

```
        cout << a [i] << "\n";
    }
  }
};

void main ()
{
  clrscr ();
  Stack s;
  do
  {
    cout << "\n Enter choice:";
    cout << "\n 1. push 2. pop s. display:";
    cin >> n;
    Switch (n)
    {
      case 1:
        s. push ();
        break;
      case 2:
        s. pop ();
        break;
      case 3 :
        s. display ();
```

Output :-

Enter choice :

1. push 2. pop 3 display 1

Enter item 50

Enter choice : 1

1. push 2. pop 3. display 1

Enter item 100

Enter choice : 2

1. push 2. pop 3. display 2

popped 100.

Enter choice: 3

1. push 2. pop 3. display 3 :

50.

```
        break;
    case 4:
        exit(0);
    }
}
while(n!=4)
    getch();
}
```

Result:

Thus the stack operation using array is got.

8. Queue using pointer

Aim:

To Implement the Queue operation using pointer.

program coding:

```
#include <iostream.h>
#include <conio.h>
#include <stdio.h>
class node
 {
   int data {
    node* next;
   public:
    node ()
    {
     data =0; next=0
    }
     friend class queue;
};
```

```
class Queue
{
  node* front; *rear;
  public:
    Queue ()
    {
     front =rear =0;
     }
    void insert (int x);
    void remove ();
     void display ();
};
void Queue :: insert (int x)
  {
   node* temp = new node;
   temp->data =x;
   if (rear ==0)
     front =rear =temp;
   else
   {rear->next =temp;
     rear =temp;
     }
 }
```

```
void Queue :: remove ()
{
 if (front ==0)
  cout << "\n Queue is empty.";
 else
 {
  cout << "Deleted" << front->data;
   front = front->next;
  }
}
void queue :: display ()
 {
 for(node *temp=front ;temp!= 0;temp= temp->next)
   cout << temp->data << "\n";
  }
 Void main ()
  {
   Int item, c;
    clrscr ();
    Queue q;
    do
   {
   cout << "\n1.Insert 2. delete 3. display
                                  4 end ";
```

Output

1. Insert 2. delete 3. display 4. exit 1

Enter item 700

1. Insert 2. delete 3. display 4. exit 1

Enter item 800

1. Insert 2. delete 3. display 4. exit 2

deleted 700

1. Insert 2. delete 3. display 4. exit 3

800.

```
cin >> c;
Switch (c)
{
 case 1:
  cout << "\n Enter item";
  cin >> item;
    q.insert (item);
    break;
 case 2:
    q.remove ();
    break;
 case 3:
    q.display ();
     break;
 default:
      exit (0);
 }
 getch ();
}
 while (1);
}
```

Result :

Thus the queue operations using pointer is implemented.

9. Queue using Array

Aim:

To implement the Queue operations using array.

program coding:

```
#include<iostream.h>
#include<conio.h>
#include<stdlib.h>
class Queue
{
  int front,rear;
  int q[5];
  public:
   Queue()
   {
     front=rear=0;
   }
   void Enqueue(int x)
   {
    if(year>4)
```

```
cout << "\n Queue is full";

else
   q [rear ++] =x;
 }
void dequeue ()
 {
   if
   cout <<"\n Queue is empty";
   else
    cout <<" \n Deletid
 }
Void display ()
 {
   for (int i = front ; i
    cout <<"\n
   }
}
Void main ()
{
 int item , ch, n;
 clrscr ();
 Queue q;
```

```
do
{
cout << "\n 1. Enqueue 2. dequeue 3. display
      4. exit";
cin >> ch;
Switch (ch)
{
 Case 1:
   cout << "\n Enter Item to Insert";
   cin >> Item;
   q. Enqueue (Item);
   break;
 Case 2:
   q. dequeue ();
   break;
 Case 3:
   q. display ();
   break;
 Case 4:
   exit (0);
}
 while (n != 4);
```

Output:

1. Enqueue. 2. deleted 3. Display 4. exit 1

Enter Item to Insert 300

1. Enqueue 2. deleted 3. display 4. exit 1

Enter Item to Insert 750.

1. Enqueue 2. deleted 3. display 4. exit 2

Deleted 300.

1. Enqueue 2. deleted 3. display 4. exit 3

750.

getch();

Result:

Thus the queue operations using array is Implemented.

10. Inorder Traversal

Aim:

To traverse the binary tree using Inorder Traversal.

Program Coding:

```
#Include<iostream.h>
#Include<conio.h>
Class node
{
  Public :
    node* LE;
    Int dab;..
    node* rc;
    node ()
    {
      Lc = rc = 0;
    }
};
Void Inorder (node* t)
  {
```

```
if (t != 0)
{
Preorder (t->lc);
Cout << t->data << "\t";
Preorder (t->rc);
}
}
void main ()
{
clrscr ();
Cout << "\n Enter the data for root:";
Int data;
Node* root;
root = new node;
Cin >> root->data;
Cout << "\n Enter data for left child:";
Node* temp = new node;
Cin >> temp->data;
root->lc = temp;
Cout << "\n Enter data for right child:";
Node * temp1 = new node;
```

Output:

Enter the data for a root : 70

Enter the data for a left child : 40

Enter the data for a right child : 90

40 70 90.

```
cin >>temp1 = new node;
root ->rc = temp1;
inorder (root);
getch();
}
```

Result:

Thus the binary tree is traversed using Inorder Traversal

11. Preorder Traversal

Aim:

To traverse the binary tree using preorder Traversal.

Program Coding:

```
#include <iostream.h>
#include<conio.h>
Class node
{
 Public:
 node* Lc;
  int data;
  node* rc;
  node()
  {
    Lc=rc=0;
  }
};
Void preorder (node* t)
```

```
{
  if (t != 0)
  {
    cout << t->data << "\t";
    preorder (t->lc);
    preorder (t->rc);
  }
}
void main ()
{
  clrscr ();
  cout << "\n Enter the data for root:";
  int data;
  node *root;
  root = new node;
  cin >> root->data;
  cout << "\n Enter data for left child:";
  node * temp = new node;
  cin >> temp->data;
```

Output:

Enter the data for root : 70

Enter data for left Child : 22

Enter data for right Child : 84

70 22 84.

```
cout << " \n Enter data for deft child:";
node* temp = new node;
cin >> temp -> data;
root -> lc = temp;
cout << " \n Enter data for right child:";
node* temp1 = new node;
cin >> temp1 -> data;
root -> rc = temp1;
preorder (root);
getch();
}
```

Result:

Thus the binary tree is traversed using preorder Traversal.

12. postorder Traversal

Aim:

To traverse the binary tree using postorder Traversal.

Program Coding:

```
#include<iostream.h>
#include<conio.h>
class node
{
public:
node* lc;
int data;
node* rc;
node()
{
lc=rc=0;
}
};
void postorder(node*);
```

```
{
 ff(t!=0)
 {
  postorder(t->LC);
  postorder(t->rc);
  Cout << t->data << "\t";
 }
}
void main()
{
  clrscr();
  Cout << "\n Enter the data for root :";
  Int data;
  node* root;
  root = new node;
  Cin >> root->data;
  Cout << "\n Enter the data for left child :";
  Node * temp = new node;
  Cin >> temp = data;
   root    lc = temp;
  Cout << "\n Enter the data for right
                                child :";
```

Output:

Enter the data for root : 77

Enter the data for left child : 100

Enter the data for right child : 23

100 23 77.

```
node * temp1 = new node;
cin >> temp1 -> data;
root -> rc = temp1;
postorder (root);
getch ();
}
```

Result:

Thus the binary tree is traversal using postorder traversal