**Slip 20**

Q.1) Write a program to implement for finding Topological sorting and determine the time complexity for the same.

#include<iostream>

#include<time.h>

using namespace std;

void findindegree(int [10][10],int[10],int); void topological(int,int [10][10]);

int main()

{

    int a[10][10],i,j,n;

    cout<<"Enter the number of nodes:";

    cin>>n;

    cout<<"\nEnter the adjacency matrix\n";

    for(i=1;i<=n;i++)

        for(j=1;j<=n;j++)

            cin>>a[i][j];

    cout<<"\nThe adjacency matirx is:\n";

    for(i=1;i<=n;i++)

    {

        for(j=1;j<=n;j++)

        {

            cout<<a[i][j]<<"\t";

        }

        cout<<"\n";

    }

    clock\_t s,e,z;

    s=clock();

    topological(n,a);

    e=clock();

    z=e-s;

    cout<<"\nTime taken:"<<z/CLOCKS\_PER\_SEC<<" seconds";

}

void findindegree(int a[10][10], int indegree[10],int n)

{

    int i,j,sum;

    for(j=1;j<=n;j++)

    {

        sum=0;

        for(i=1;i<=n;i++)

        {

            sum = sum + a[i][j];

        }

        indegree[j] = sum;

    }

}

void topological(int n,int a[10][10])

{

    int k,top,t[100],i,stack[20],u,v,indegree[20];

    k=1;

    top=-1;

    findindegree(a,indegree,n);

    for(i=1;i<=n;i++)

    {

        if(indegree[i]=0)

        {

            stack[++top]=i;

        }

    }

    while(top!=-1)

    {

        u=stack[top--];

        t[k++]=u;

        for(v=1;v<=n;v++)

        if(a[u][v]==1)

        {

            indegree[v]--;

            if (indegree[v]=0)

            {

                stack[++top]=v;

            }

        }

    }

    cout<<"\nTopological sequence is\n";

    for(i=1;i<=n;i++)

    cout<<t[i]<<"\t";

}

Q.2) Write a program to solve N Queens Problem using Backtracking.

#include<iostream>

#include<math.h>

using namespace std;

void nqueens(int);

int place(int[],int);

void printsolution(int,int[]);

int main()

{

    int n;

    cout<<"Enter the no. of queens:";

    cin>>n;

    nqueens(n);

}

void nqueens(int n)

{

    int x[10],count=0,k=1;

    x[k]=0;

    while(k!=0)

    {

        x[k]=x[k]+1;

        while(x[k]<=n&&(!place(x,k)))

            x[k]=x[k]+1;

        if(x[k]<=n)

        {

            if(k==n)

            {

                count++;

                cout<<"\nSolution"<<count<<"\n";

                printsolution(n,x);

            }

            else

            {

                k++;

                x[k]=0;

            }

        }

        else

        {

            k--;

        }

    }

}

int place(int x[],int k)

{

    int i;

    for(i=1;i<k;i++)

        if(x[i]==x[k]||(abs(x[i]-x[k]))==abs(i-k))

            return 0;

    return 1;

}

void printsolution(int n,int x[])

{

    int i,j;

    char c[10][10];

    for(i=1;i<=n;i++)

    {

        for(j=1;j<=n;j++)

        c[i][j]='X';

    }

    for(i=1;i<=n;i++)

        c[i][x[i]]='Q';

    for(i=1;i<=n;i++)

    {

        for(j=1;j<=n;j++)

        {

            cout<<c[i][j]<<"\t";

        }

        cout<<"\n";

    }

}