**Assignment No:-**

**Assignment Name**:- **Write a program to find all solutions for 8-queen problem using backtracking.**

**Name:- WARKE PURVA DILIP.**

**Roll No:- 136.**

----------------------------------------------------------------------------------------------------------------------------------------------------

#include<iostream.h>

#include<conio.h>

#include<stdlib.h>

#include<math.h>

class QUEEN

{

private:

int X[10],n,count;

public:

QUEEN(int);

void NQUEEN(int);

int PLACE(int);

void GET\_RESULT();

};

QUEEN::QUEEN(int par)

{

n=par;

for(int i=1;i<=n;i++)

X[i]=0;

count=0;

}

void QUEEN::NQUEEN(int n)

{

int k=1;

X[k]=0;

//--- false - tried all possible solutions ---

while(k>0)

{

X[k]=X[k]+1;

//----queen is in board && not attacking

while( X[k]<=n && !PLACE(k) )

{

X[k]=X[k]+1;

}

if(X[k]<=n) // queen is placed

{

if(k==n)//last queen

{

count++;

GET\_RESULT();

}

else // next queen

{

k=k+1;

X[k]=0;

}

}

else // back track

{

k = k-1;

}

}

cout<<endl<<"No of sol : "<<count;

}

int QUEEN::PLACE(int k)

{

for(int i=1;i<=k-1;i++)

{ // match col && diagonal with prev queens

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

return 0;

}

return 1;

}

void QUEEN::GET\_RESULT()

{

cout<<endl;

for(int i=1;i<=n;i++) // row

{

cout<<" | ";

for(int j=1;j<=n;j++) // col

{

if(j==X[i])

cout<<X[i]<<" | ";

else

cout<<X[i]<<" | ";

}

cout<<endl;

}

}

void main()

{

int n;

clrscr();

cout<<"\nEnter no of Queens : ";

cin>>n;

QUEEN obj(n);

obj.NQUEEN(n);

getch();

}