LAB 5: BACKTRACKING APPROACH

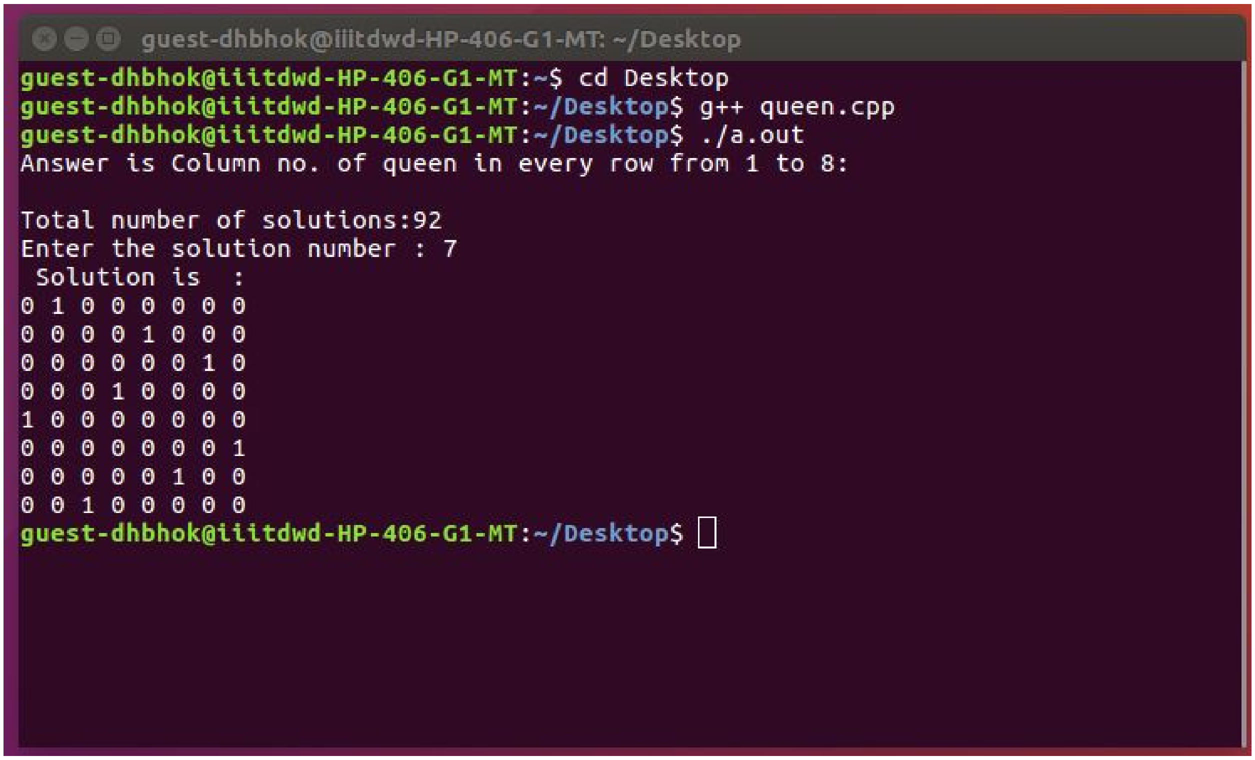
Date : 4.4.18

Submitted By : Abhineet Singh

1. ***Write a program to implement 8-queen problem using backtracking approach***

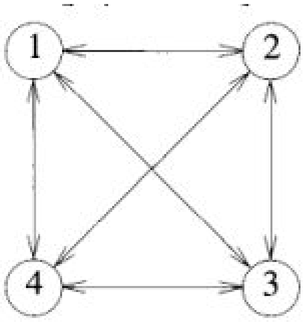
```
#include<iostream>
#include <bits/stdc++.h>
#include<cmath>


using namespace std;

vector< vector<int> >  solution;

int  x[8];
int cnt = 0;
int place(int k,int i){
        for(int j = 0;j<k;j++){
        if((x[j]==i) || abs(x[j]-i) == abs(j-k)){
        return 0;
        }
        }
        return 1;
}

int queen(int k,int n){
        for(int i = 0; i<8; i++){
        if(place(k,i)){
        x[k] = i;
        if(k==n-1){
                cnt++;
        vector<int>  sol;
                for(int z=0; z<8; z++){
                sol.push_back(x[z]);
                }
        solution.push_back(sol);

        }
```

```
    else( queen(k+1,n));
    }

    }
}



int main(){
    int board[8][8]=  {0};
    int k=0, n=8,num;
    cout<<"Answer is Column no. of queen in every row from 1 to 8:"<<endl;

    queen(k,n);
    cout<<"\nTotal number of solutions:"<<cnt<<endl;
    cout << "Enter the solution number : ";
    cin >> num ;
    cout <<" Solution is  : \n";
    int i=0;
    for(int j=0 ; j<8;j++){
    board[j][solution[num-1][i]]=1;
    i++;
    }

 for(int i=0 ; i<8;i++){
    for(int j=0;j<8;j++){
    cout << board[i][j]<< " " ;
    }
    cout<<endl;

  }
    return 0;
}
```



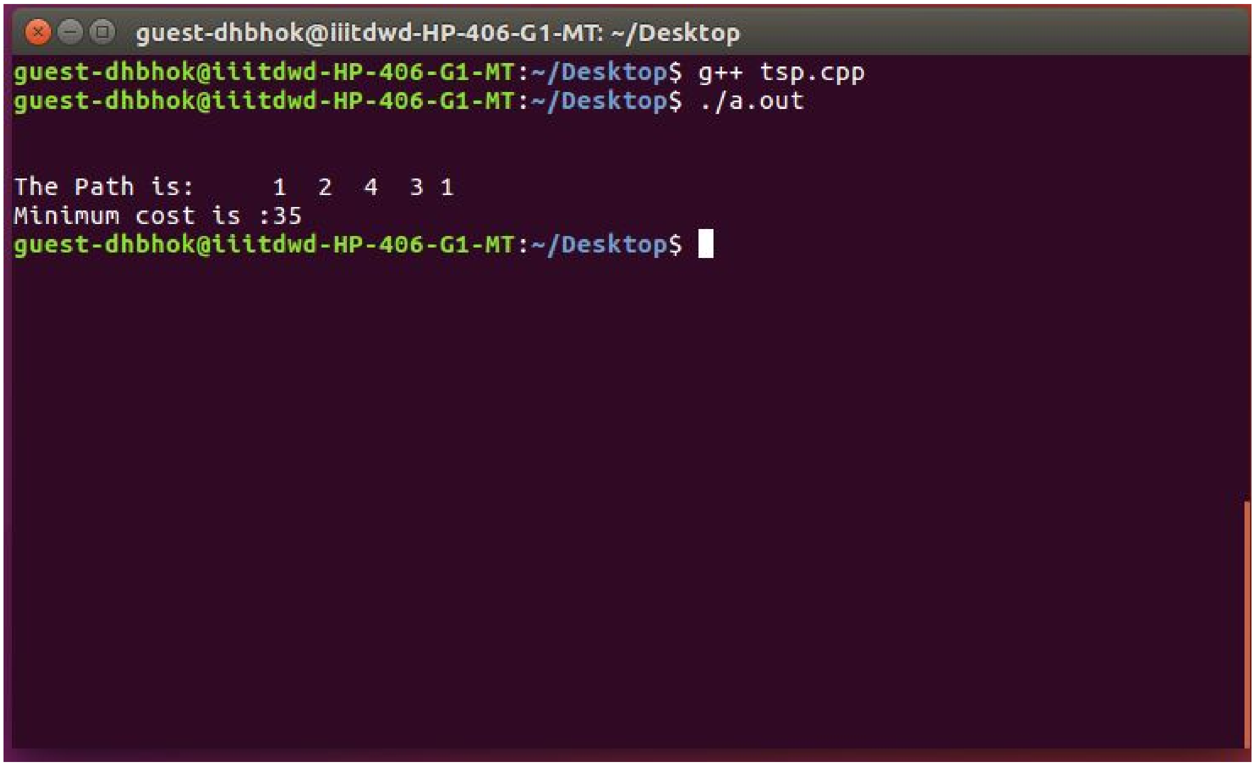
```
guest-dhbhok@iiitdwd-HP-406-G1-MT:~$ cd Desktop
guest-dhbhok@iiitdwd-HP-406-G1-MT:~/Desktop$ g++ queen.cpp
guest-dhbhok@iiitdwd-HP-406-G1-MT:~/Desktop$ ./a.out
Answer is Column no. of queen in every row from 1 to 8:

Total number of solutions:92
Enter the solution number : 7
 Solution is  :
0 1 0 0 0 0 0 0
0 0 0 0 1 0 0 0
0 0 0 0 0 0 1 0
0 0 0 1 0 0 0 0
1 0 0 0 0 0 0 0
0 0 0 0 0 0 0 1
0 0 0 0 0 1 0 0
0 0 1 0 0 0 0 0
guest-dhbhok@iiitdwd-HP-406-G1-MT:~/Desktop$
```

2. ***Write a program to implement travelling salesman problem using dynamic programming for a directed graph with adjacency cost matrix.***



$$\begin{bmatrix} 0 & 10 & 15 & 20 \\ 5 & 0 & 9 & 10 \\ 6 & 13 & 0 & 12 \\ 8 & 8 & 9 & 0 \end{bmatrix}$$

```
#include<iostream>
#include<bits/stdc++.h>

using namespace std ;
int visited[4]={0},n=4,cost=0;
int a[4][4] ={{0,10,15,20},{5,0,9,10},{6,13,0,12},{8,8,9,0}};


int least(int c)
{
        int i,nc=INT_MAX;
        int min=INT_MAX,kmin;
        for(i=0;i < n;i++)
        {
        if((a[c][i]!=0)&&(visited[i]==0))
        if(a[c][i] < min)
        {
                min=a[i][0]+a[c][i];
                kmin=a[c][i];
                nc=i;
        }
        }
        if(min!=INT_MAX)
        cost+=kmin;
        return nc;
}


void mincost(int city)
{
        int i,ncity;
        visited[city]=1;
        cout << " " <<city+1 <<" ";
        ncity=least(city);
        if(ncity==INT_MAX)
        {
        ncity=0;
        cout << ncity+1;
        cost+=a[city][ncity];
        return;
        }
        mincost(ncity);
}




int main()
{


        cout << "\n\nThe Path is:\t";
        mincost(0);
        cout << "\nMinimum cost is :" << cost<<endl;
}
```



```
guest-dhbhok@iiitdwd-HP-406-G1-MT:~/Desktop$ g++ tsp.cpp
guest-dhbhok@iiitdwd-HP-406-G1-MT:~/Desktop$ ./a.out


The Path is:     1  2  4  3 1
Minimum cost is :35
guest-dhbhok@iiitdwd-HP-406-G1-MT:~/Desktop$ 
```