G. H. Raisoni College Of Engineering And Management, Wagholi Pune

2021- 2022

Assignment no :- 11

Department	CE [SUMMER 2022 (Online)]		
Term / Section	III/B	Date Of **submission**	13-12-2021
Subject Name /Code	Data Structures and Algorithms/ UCSL201/UCSP201		
Roll No.	SCOB77	Name	Pratham Rajkumar pitty
Registration Number	2020AC0E1100107		

Experiment No 11

E

Aim :→ A classic problem that can be solved by backtracking is called the ~~q~~ eight Queens problem, which comes from the game of chess. The chess board consist of 64 square arranged in an 8 by 8 grid. The board normally alternates between black and white square, but this is not relevant for the present problem. The queen can move as far as she wants in any direction, as long as she follows a ~~string~~ straight line, vertically, horizontally, or diagonally. write C++ program for generating all possible configurations for 4-queen's problem.

Theory :→ This problem is to find an arrangement of N queens ~~can attack in any direction~~ as on a chess board, such that no queen can attack any other queens on the board. The chess queens can attack in any direction as horizontal, vertical, horizontal and diagonal way. A binary matrix is used to display the positions of N queens, where no queens can attack other queens.

Algorithm

isValid(board, row, col)

Input: The chess board, row and the column of the board

Output:→ True when placing a queen in row and place position is a valid or not.

```
Begin
  if there is a queen at the left of current col, then
  ~~then~~ return false
  if there is a queen at the left upper diagonal, then
    return false
  if there is a queen at the left lower diagonal, then
    return false.
  return true // otherwise it is valid place.
End.
```

SolveNQueen (board, col)

Input - The chess board, the col where the queen is
        trying to be placed.

Output - The position matrix where queens are placed

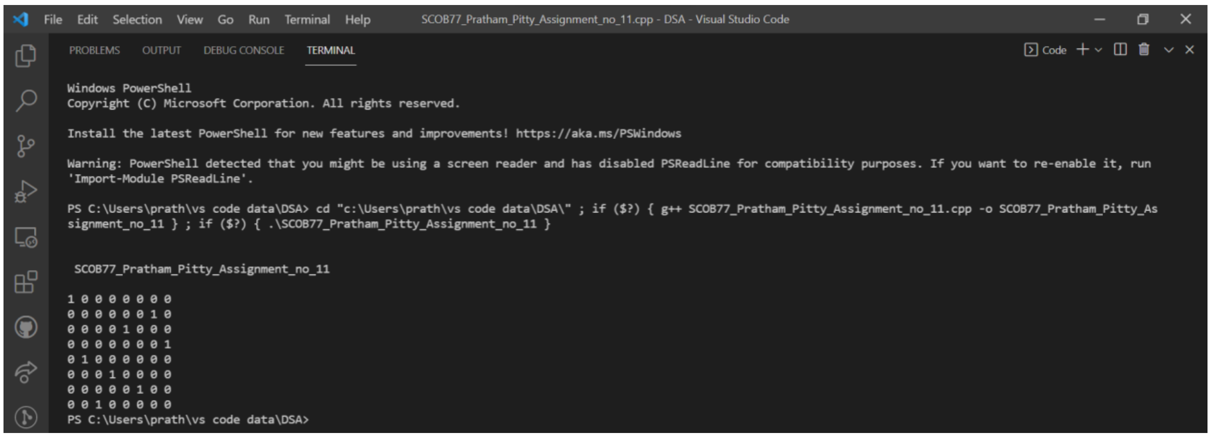
```
Begin
   if all colums are filled, them
   return true.
   for each row of the board, do
   if isvalid ( board, i, col), then,
   set queen at place (i, col) in the board
   if SolveNQueen (board, col+1) = true, then
     return ture
   otherwise remove queen froon place (i, col) from
                                                board.
   done
   return false
End
```

Program code :-

```
#include <iostream>

using namespace std;

#define N 8


void printBoard(int board[N][N])

{

   for (int i = 0; i < N; i++)

   {

      for (int j = 0; j < N; j++)

         cout << board[i][j] << " ";

      cout << endl;

   }

}


bool isValid(int board[N][N], int row, int col)

{

   for (int i = 0; i < col; i++) // check whether there is queen in the left or not

      if (board[row][i])

         return false;

   for (int i = row, j = col; i >= 0 && j >= 0; i--, j--)

      if (board[i][j]) // check whether there is queen in the left upper diagonal or not

         return false;

   for (int i = row, j = col; j >= 0 && i < N; i++, j--)

      if (board[i][j]) // check whether there is queen in the left lower diagonal or not

         return false;

   return true;

}


bool solveNQueen(int board[N][N], int col)
```

```
{
   if (col >= N) // when N queens are placed successfully
      return true;
   for (int i = 0; i < N; i++)
   { // for each row, check placing of queen is possible or not
      if (isValid(board, i, col))
      {
         board[i][col] = 1;          // if validate, place the queen at place (i, col)
         if (solveNQueen(board, col + 1)) // Go for the other columns recursively
            return true;

         board[i][col] = 0; // When no place is vacant remove that queen
      }
   }
   return false; // when no possible order is found
}

bool checkSolution()
{
   int board[N][N];
   for (int i = 0; i < N; i++)
      for (int j = 0; j < N; j++)
         board[i][j] = 0; // set all elements to 0

   if (solveNQueen(board, 0) == false)
   { // starting from 0th column
      cout << "Solution does not exist";
      return false;
   }
   printBoard(board);
   return true;
}


int main()
{
   cout << "\n\n SCOB77_Pratham_Pitty_Assignment_no_11 \n\n";
   checkSolution();
}
```

Output :-


File Edit Selection View Go Run Terminal Help
SCOB77_Pratham_Pitty_Assignment_no_11.cpp - DSA - Visual Studio Code
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
Code

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

Warning: PowerShell detected that you might be using a screen reader and has disabled PSReadLine for compatibility purposes. If you want to re-enable it, run
'Import-Module PSReadLine'.

PS C:\Users\prath\vs code data\DSA> cd "c:\Users\prath\vs code data\DSA\" ; if ($?) { g++ SCOB77_Pratham_Pitty_Assignment_no_11.cpp -o SCOB77_Pratham_Pitty_As
signment_no_11 } ; if ($?) { .\SCOB77_Pratham_Pitty_Assignment_no_11 }


 SCOB77_Pratham_Pitty_Assignment_no_11

1 0 0 0 0 0 0 0
0 0 0 0 0 0 1 0
0 0 0 0 1 0 0 0
0 0 0 0 0 0 0 1
0 1 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 0
PS C:\Users\prath\vs code data\DSA>
```