Experiment 9 (Backtracking)

Student Name: Lipakshi **UID: 20BCS5082**

Branch: BE CSE **Section/Group: 607 / B**

Semester: 5th **Date of performance: 02.11.22**

Subject: Competitive Coding **Subject Code: 20CSP_314**

1. Aim/Overview of the Practical:

a. Queens on n board.
b. A tryst with chess.

2. Task to be done / Which logistics used:

a. You have an *N* * *M* chessboard on which some squares are blocked out. In how many ways can you place one or more queens on the board, such that, no two queens attack each other? Two queens attack each other, if one can reach the other by moving horizontally, vertically, or diagonally without passing over any blocked square. At most one queen can be placed on a square. A queen cannot be placed on a blocked square.

b. Given a chess board having N×N cells, you need to place *N* queens on the board in such a way that no queen attacks any other queen.

3. Steps for experiment/practical/Code:

a. Queens on n board

```
LL rec(int index, int lastv, int lastl, int lastr) {
if (index == N) return 1;

LL &ans = P[index][lastv][lastl][lastr];

if (ans != -1) return ans;
```

```
    ans = 0;
    for(int jk=0;jk<js[M-1].size();++jk) {
        int j = js[M-1][jk];
        bool good = true;
        vector<int> pos;
        for(int k = 0; k < M; ++k)
            if ((j>>k)&1)
                if (((board[index]>>k)&1) && !((lastv>>k)&1) && !((lastl>>k)&1) &
& !((lastr>>k)&1))
                    pos.push_back(k)
                ; else
                    good =
        false; if (!good)
        continue;
        for(int i = 0; i + 1 < pos.size(); ++i) {
            bool found = false;
            for(int k = pos[i]+1;
                k<pos[i+1];++k) if
                (!((board[index]>>k)&1))
                    found = true;
            if (!found) good = false;
        }
        if (!good) continue;
        ans += rec(index + 1, (board[index]&lastv)|j,
                    ((j<<1)|((board[index]&lastl)<<1))&((1<<M)-1),
                    (j>>1)|((board[index]&lastr)>>1));
    }
    return ans%mod;

}
```

b. A tryst with chess:

```
#include <iostream>
using namespace
std;
bool issafe(int **arr, int x, int y, int n)
{
   for (int row = 0; row < x; row++)
   {
      if (arr[row][y] == 1)
      {
         return 0;
      }
   }
   int row =
   x; int col =
   y;
   while (row >= 0 && col >= 0)
   {
      if (arr[row][col] == 1)
      {
         return 0;
      }
      row--;
      col--;
   }
   row = x;
   col = y;



   while (row >= 0 && col < n)
   {
      if (arr[row][col] == 1)
      {
         return 0;
      }
      row--;
```

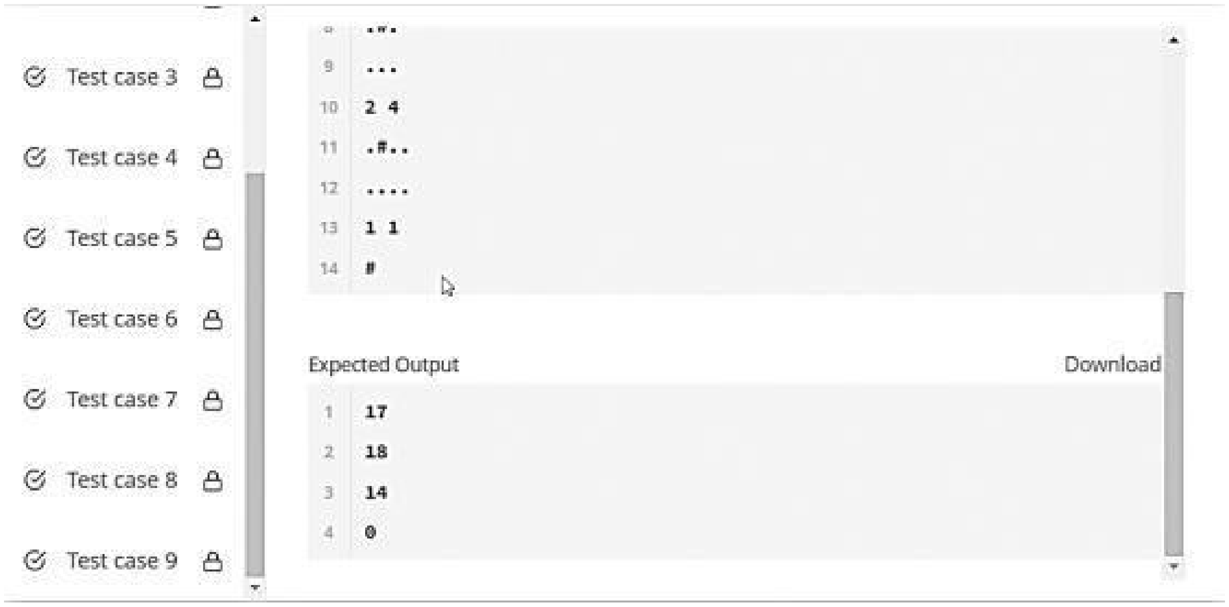
```
        col++;
    }
    return 1;
}
bool nqueen(int **arr, int x, int n)
{
    if (x >= n)
    {
        return 1;
    }
    for (int col = 0; col < n; col++)
    {
        if (issafe(arr, x, col, n))
        {
            arr[x][col] = 1;
            if (nqueen(arr, x + 1, n))
            {
                return 1;
            }
            arr[x][col] = 0;
        }
    }
    return 0;
}
int main()
{
    int n;
    cin >> n;
    int **arr = new int
    *[n]; for (int i = 0; i <
    n; i++)
    {
        arr[i] = new int[n];
        for (int j = 0; j < n; j++)
        {
            arr[i][j] = 0;
        }
    }
    if (nqueen(arr, 0, n))



    {
        for (int i = 0; i < n; i++)
        {
                for (int j = 0; j < n; j++)
                {
                    cout << arr[i][j] << " ";
                }
                cout << endl;
            }
        }
        else
        {
            cout << "Not possible" << endl;
        }
        return 0;
    }
```

Result/Output/Writing Summary:

a. Queens on n board:



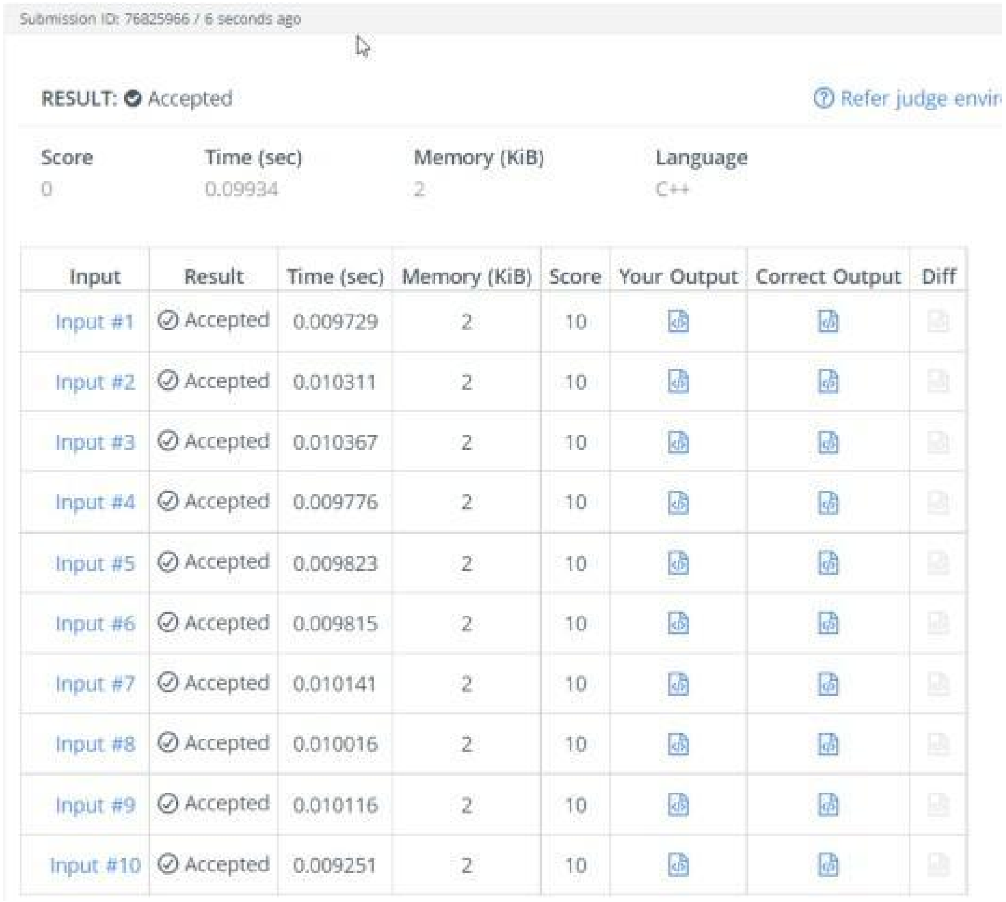
Test case 3
Test case 4
Test case 5
Test case 6
Test case 7
Test case 8
Test case 9

```
.#.
...
2 4
.#..
....
1 1
#
```

Expected Output
Download

```
17
18
14
0
```

b. A tryst with chess:



Submission ID: 76825966 / 6 seconds ago

RESULT: Accepted

Refer judge envir

Score	Time (sec)	Memory (KiB)	Language
0	0.09934	2	C++

Input	Result	Time (sec)	Memory (KiB)	Score	Your Output	Correct Output	Diff
Input #1	Accepted	0.009729	2	10			
Input #2	Accepted	0.010311	2	10			
Input #3	Accepted	0.010367	2	10			
Input #4	Accepted	0.009776	2	10			
Input #5	Accepted	0.009823	2	10			
Input #6	Accepted	0.009815	2	10			
Input #7	Accepted	0.010141	2	10			
Input #8	Accepted	0.010016	2	10			
Input #9	Accepted	0.010116	2	10			
Input #10	Accepted	0.009251	2	10			

<u>Learning outcomes (What I have learnt):</u>

a. Learnt about backtracking.
b. Got an overview of the implementation of recursion.
c. Get to know about crucial test cases.