

Name: Ashmit Thawait

Roll No: 102203790

Group: 2CO17

---

## Lab Assignment 4

### Q1. N-queen Problem

```
1  #include <bits/stdc++.h>
2  using namespace std;
3  #define N 4
4
5  int ld[30] = { 0 };
6  int rd[30] = { 0 };
7  int cl[30] = { 0 };
8
9  void printSolution(int board[N][N])
10 {
11     for (int i = 0; i < N; i++) {
12         for (int j = 0; j < N; j++)
13             cout << " " << (board[i][j]==1?"Q":".") << " ";
14         cout << endl;
15     }
16 }
17
18 bool solveNQueUtil(int board[N][N], int col)
19 {
20     if (col >= N)
21         return true;
22
23     for (int i = 0; i < N; i++) {
24         if ((ld[i - col + N - 1] != 1 && rd[i + col] != 1) && cl[i] != 1) {
25             board[i][col] = 1;
26             ld[i - col + N - 1] = rd[i + col] = cl[i] = 1;
27
28             if (solveNQueUtil(board, col + 1))
29                 return true;
30
31             board[i][col] = 0;
32             ld[i - col + N - 1] = rd[i + col] = cl[i] = 0;
33         }
34     }
35     return false;
36 }
37
38 }
```

```

40  bool solveNQ()
41  {
42      int board[N][N] = { { 0, 0, 0, 0 },
43                          { 0, 0, 0, 0 },
44                          { 0, 0, 0, 0 },
45                          { 0, 0, 0, 0 } };
46
47      if (solveNQUtil(board, 0) == false) {
48          cout << "Solution does not exist";
49          return false;
50      }
51
52      printSolution(board);
53      return true;
54  }
55
56  int main()
57  {
58      solveNQ();
59      return 0;
60  }

```

```

PS D:\DAA Assignments\Assignment 4> g++ n_queen.cpp
PS D:\DAA Assignments\Assignment 4> ./a.exe

```

```

. . Q .
Q . . .
. . . Q
. Q . .

```

## Q2. Sum of subsets

```
1  #include <bits/stdc++.h>
2  using namespace std;
3
4  bool flag = 0;
5  void PrintSubsetSum(int i, int n, int set[], int targetSum, vector<int>& subset)
6  {
7      if (targetSum == 0) {
8          flag = 1;
9          cout << "[ ";
10         for (int i = 0; i < subset.size(); i++) {
11             cout << subset[i] << " ";
12         }
13         cout << "]";
14         return;
15     }
16 }
17
18 if (i == n) {
19     return;
20 }
21
22 PrintSubsetSum(i + 1, n, set, targetSum, subset);
23
24 if (set[i] <= targetSum) {
25     subset.push_back(set[i]);
26
27     PrintSubsetSum(i + 1, n, set, targetSum - set[i],
28                   subset);
29
30     subset.pop_back();
31 }
32 }
33 }
```

```
35 int main()
36 {
37     int set[] = { 1, 2, 1 };
38     int sum = 3;
39     int n = sizeof(set) / sizeof(set[0]);
40     vector<int> subset;
41     cout << "Output 1:" << endl;
42     PrintSubsetSum(0, n, set, sum, subset);
43     cout << endl;
44     flag = 0;
45
46     int set2[] = { 3, 34, 4, 12, 5, 2 };
47     int sum2 = 30;
48     int n2 = sizeof(set) / sizeof(set[0]);
49     vector<int> subset2;
50     cout << "Output 2:" << endl;
51     PrintSubsetSum(0, n2, set2, sum2, subset2);
52     if (!flag) {
53         cout << "There is no such subset";
54     }
55
56     return 0;
57 }
```

```
PS D:\DAA Assignments\Assignment 4> g++ SumOfSubsets.cpp
PS D:\DAA Assignments\Assignment 4> ./a.exe
Output 1:
[ 2 1 ][ 1 2 ]
Output 2:
There is no such subset
PS D:\DAA Assignments\Assignment 4> █
```

### Q3. Graph coloring

```
1  #include <bits/stdc++.h>
2  using namespace std;
3  #define V 4
4
5  void printSolution(int color[]);
6
7  bool isSafe(int v, bool graph[V][V], int color[], int c)
8  {
9      for (int i = 0; i < V; i++)
10         if (graph[v][i] && c == color[i])
11             return false;
12
13     return true;
14 }
15
16 bool graphColoringUtil(bool graph[V][V], int m, int color[],
17                        int v)
18 {
19     if (v == V)
20         return true;
21
22     for (int c = 1; c <= m; c++) {
23
24         if (isSafe(v, graph, color, c)) {
25             color[v] = c;
26
27             if (graphColoringUtil(graph, m, color, v + 1)
28                 == true)
29                 return true;
30
31             color[v] = 0;
32         }
33     }
34     return false;
35 }
```

```

37 bool graphColoring(bool graph[V][V], int m)
38 {
39     int color[V];
40     for (int i = 0; i < V; i++)
41         color[i] = 0;
42
43     if (graphColoringUtil(graph, m, color, 0) == false) {
44         cout << "Solution does not exist";
45         return false;
46     }
47
48     printSolution(color);
49     return true;
50 }
51
52 void printSolution(int color[])
53 {
54     cout << "Solution Exists:"
55         << " Following are the assigned colors"
56         << "\n";
57     for (int i = 0; i < V; i++)
58         cout << " " << color[i] << " ";
59
60     cout << "\n";
61 }
62
63 int main()
64 {
65     bool graph[V][V] = {
66         { 0, 1, 1, 1 },
67         { 1, 0, 1, 0 },
68         { 1, 1, 0, 1 },
69         { 1, 0, 1, 0 },
70     };
71
72     int m = 3;
73     graphColoring(graph, m);
74     return 0;
75 }

```

```

PS D:\DAA Assignments\Assignment 4> g++ .\GraphColouring.cpp
PS D:\DAA Assignments\Assignment 4> ./a.exe
Solution Exists: Following are the assigned colors
 1 2 3 2
PS D:\DAA Assignments\Assignment 4> █

```