ASSIGNMENT NO:- 02
ROLL NO :- 33252

```
#include <bits/stdc++.h>
using namespace std;

bool place(int s[], int row, int j, int n) {
    for (int k = 0; k < row; k++) {
        // Check if the same column or not
        if (s[k] == j) {
            return false;
        }
        // Check if the same diagonal or not
        if (abs(s[k] - j) == abs(k - row)) {
            return false;
        }
    }
    return true;
}

void print(int store_col[], int n, string message = "") {
    if (!message.empty()) {
        cout << message << endl;
    }
    for (int i = 0; i < n; i++) {
        for (int j = 0; j < n; j++) {
            if (store_col[i] == j)
                cout << "Q ";
            else
                cout << ". ";
        }
        cout << endl;
    }
    cout << endl;
}

void nqueen(int store_col[], int n, int row) {
    // Print intermediate state of the board
    print(store_col, n, "Current Iteration:");

    if (row == n) {
        cout << "Next Iteration: Solution Found" << endl;
        print(store_col, n);
        return;
    }

    for (int i = 0; i < n; i++) {
        if (place(store_col, row, i, n)) {
            store_col[row] = i;
            nqueen(store_col, n, row + 1);
            store_col[row] = -1; // Backtrack
```

```
        }
    }
}

int main() {
    int n;
    cout << "Enter the number of queens to place: " << endl;
    cin >> n;

    if (n <= 3) {
        cout << "INVALID NUMBER" << endl;
    } else {
        int store_col[n];
        for (int i = 0; i < n; i++) {
            store_col[i] = -1;
        }
        nqueen(store_col, n, 0);
    }
    return 0;
}
```

OUTPUT

```
Enter the number of queens to place:
4
Current Iteration:
. . . .
. . . .
. . . .
. . . .

Current Iteration:
Q . . .
. . . .
. . . .
. . . .

Current Iteration:
Q . . .
. . Q .
. . . .
. . . .

Current Iteration:
Q . . .
. . . Q
. . . .
. . . .

Current Iteration:
Q . . .
```

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

Current Iteration:
. Q . .
. . . .
. . . .
. . . .

Current Iteration:
. Q . .
. . . Q
. . . .
. . . .

Current Iteration:
. Q . .
. . . Q
Q . . .
. . . .

Current Iteration:
. Q . .
. . . Q
Q . . .
. . Q .

Next Iteration: Solution Found
. Q . .
. . . Q
Q . . .
. . Q .

Current Iteration:
. . Q .
. . . .
. . . .
. . . .

Current Iteration:
. . Q .
Q . . .
. . . .
. . . .

Current Iteration:
. . Q .
Q . . .
. . . Q
. . . .

Current Iteration:
. . Q .
Q . . .
. . . Q
. Q . .

Next Iteration: Solution Found
. . Q .
Q . . .
. . . Q
. Q . .

Current Iteration:
. . . Q
. . . .
. . . .
. . . .

Current Iteration:
. . . Q
Q . . .
. . . .
. . . .

Current Iteration:
. . . Q
Q . . .
. . Q .
. . . .

Current Iteration:
. . . Q
. Q . .
. . . .
. . . .