Castle on the Grid

locked

Submitted a few seconds ago • Score: 1.00 Status: Accepted

✔ Test Case #0	✔ Test Case #1	✔ Test Case #2
✔ Test Case #3	✔ Test Case #4	✔ Test Case #5
✔ Test Case #6	✔ Test Case #7	✔ Test Case #8
✔ Test Case #9	✔ Test Case #10	✔ Test Case #11
✔ Test Case #12	✔ Test Case #13	

Submitted Code

Language: C++ Open in editor

```
// Sample Input

// 3
// .X.
// .X.
// ...
// 0 0 0 2
// Sample Output

// 3




#include <cmath>
#include <cstdio>
#include <vector>
#include <iostream>
#include <algorithm>

#include <queue>
using namespace std;
char mark = 'X';

int main() {
    int n;
    cin >> n;

    char c, m[n][n];
    for (int i = 0; i < n; ++i) {
        for (int j = 0; j < n; ++j) {
            cin >> c;
            m[i][j] = c;
        }
```

```
    }

    int xs, ys, xd, yd;
    cin >> xs >> ys >> xd >> yd;

    queue<int> dist;
    queue<pair<int, int>> q;
    q.push(make_pair(xs, ys));
    dist.push(0);
    m[xs][ys] = mark;

    vector<pair<int, int>> mv = { make_pair(-1, 0), make_pair(0, +1), make_pair(+1, 0), make_pair(0, -1) };

    while (!q.empty())
    {
        pair<int, int> cur = q.front();

        if (cur.first == xd && cur.second == yd)
        {
            cout << dist.front() << endl;
            break;
        }

        for (int i = 0; i < mv.size(); ++i)
        {
            int newX = cur.first + mv[i].first;
            int newY = cur.second + mv[i].second;

            while (newX >= 0 && newX < n && newY >= 0 && newY < n && m[newX][newY] != mark)
            {
                q.push(make_pair(newX, newY));
                dist.push(dist.front() + 1);
                m[newX][newY] = mark;

```

```
                newX += mv[i].first;
                newY += mv[i].second;
            }
        }

        q.pop();
        dist.pop();
    }

    return 0;
}
```