

## Knight moves (BFS in 2d Grid) — [Knight moves problem in Onlinejudge](#)

Knight এর সংক্ষেপ পথ: একটি BFS সমাধান

### সমস্যা বিবরণ (Problem Description)

এই সমস্যা হলো shortest path সমস্যা। দাবা বোর্ডের একটি নাইটকে (Knight) নির্দিষ্ট একটি স্কোয়ার থেকে অন্য স্কোয়ার পর্যন্ত পৌঁছাতে হবে। আমাদের লক্ষ্য হলো ন্যূনতম কতটি নাইট চাল লাগবে তা নির্ণয় করা।

Board Details:

- ৮×৮ স্কোয়ার (দাবা বোর্ড)
- প্রতিটি স্কোয়ার একটি নোড হিসাবে বিবেচিত
- নাইটের বৈধ চাল দুটি স্কোয়ারের মধ্যে একটি এজ তৈরি করে
- প্রতিটি চালের ওজন সমান (১)

সুতরাং, এটি একটি ওজনহীন গ্রাফ যেখানে shortest path খুঁজতে BFS (Breadth-First Search) ব্যবহার করা সবচেয়ে কার্যকর।

---

### নাইটের চাল (Knight Moves)

নাইট ৮টি ভিন্ন দিক দিয়ে চলে:

Row Change (kr) Column Change (kc)

2	1
2	-1
-2	1
-2	-1
1	2
1	-2
-1	2
-1	-2

---

### BFS ব্যবহার (Why BFS?)

BFS একটি গ্রাফ ট্রাভার্সাল অ্যালগরিদম যা level by level অনুসন্ধান করে।

- Layer 0: শুরু স্কোয়ার
- Layer 1: ১টি চাল থেকে পৌঁছানো স্কোয়ার
- Layer 2: ২টি চাল থেকে পৌঁছানো স্কোয়ার
- ...

BFS প্রথমবার যখন লক্ষ্য স্কোয়ারে পৌঁছায়, তখন সেটি ন্যূনতম চালের সংখ্যা।

### Features

- ন্যূনতম নাইট চাল বের করা

- BFS ব্যবহার করে efficient সমাধান
- ৮×৮ বোর্ডের জন্য fast execution
- সহজে অন্য বোর্ড সাইজেও extend করা সম্ভব

### Pseudocode

```

FUNCTION bfs_knight(startR, startC, endR, endC):
    N = 8
    DEFINE dist[N][N] AS ARRAY
    DEFINE color[N][N] AS ARRAY
    DEFINE Queue Q
    FOR r FROM 0 TO N-1:
        FOR c FROM 0 TO N-1:
            dist[r][c] = INFINITY
            color[r][c] = -1
    dist[startR][startC] = 0
    color[startR][startC] = 1
    ENQUEUE (startR, startC) INTO Q

    WHILE Q IS NOT EMPTY:
        (r, c) = DEQUEUE FROM Q
        IF (r, c) IS EQUAL TO (endR, endC):
            RETURN dist[r][c]
        FOR i FROM 0 TO 7:
            newR = r + kr[i]      // kr[] = {2, 2, -2, -2, 1, 1, -1, -1}
            newC = c + kc[i]      // kc[] = {1, -1, 1, -1, 2, -2, 2, -2}

            IF newR, newC IS ON BOARD AND color[newR][newC] IS -1:
                color[newR][newC] = 1
                dist[newR][newC] = dist[r][c] + 1
                ENQUEUE (newR, newC) INTO Q
            color[r][c] = 2
    RETURN -1

```