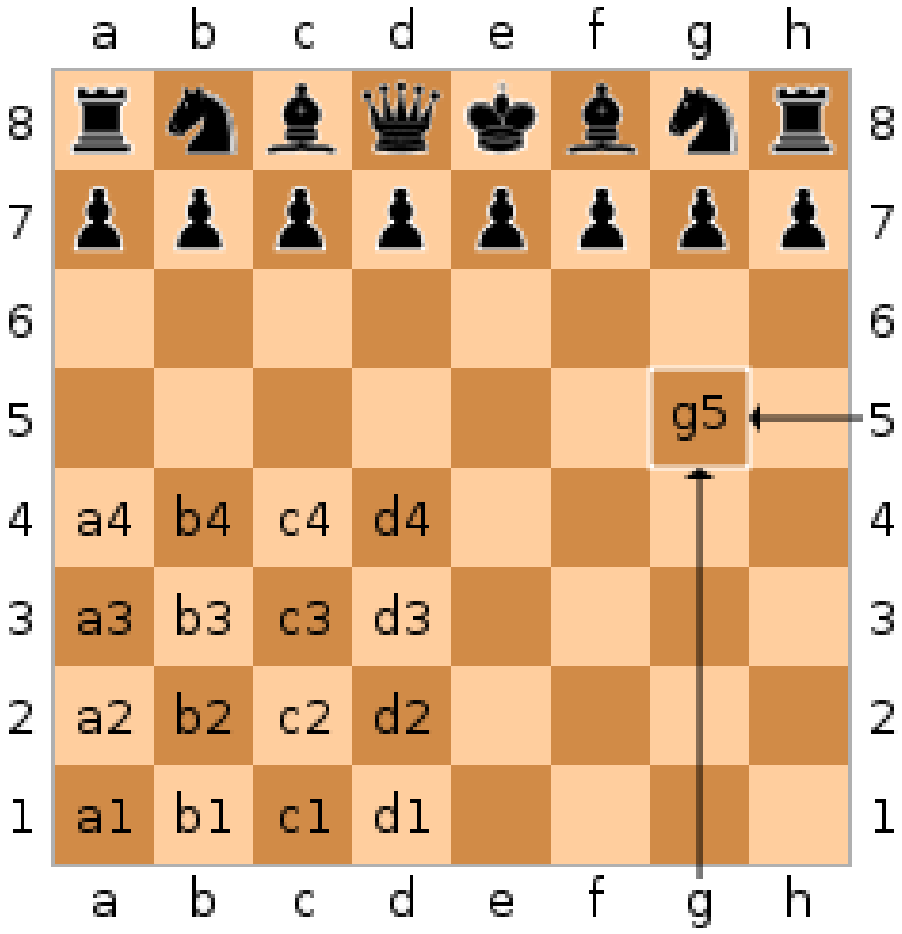
Bàn cờ vua là một bảng có `8*8` ô vuông

Mỗi ô trên bàn cờ được kí hiệu bằng `2` kí tự - `1` kí tự chữ cái và `1` kí tự số

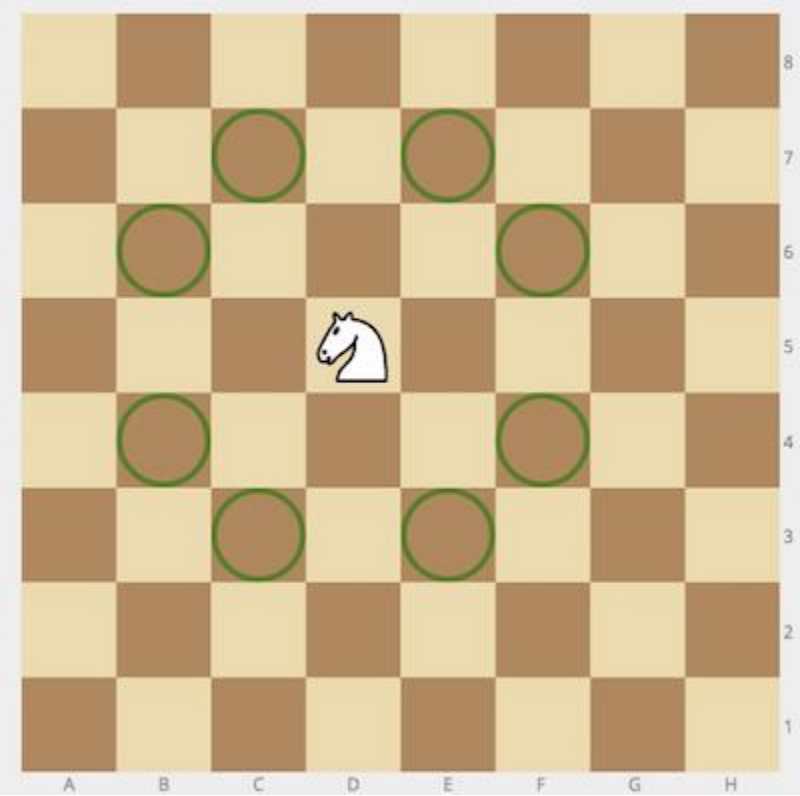
- Các cột hàng dọc được gán nhãn từ trái sang phải bằng các kí tự chữ cái từ `'a'` tới `'h'`
- Các hàng ngang được đánh số từ `1` tới `8` từ phía dưới lên trên
- Vị trí mỗi một ô trên bàn cờ được thể hiện bằng xâu có `2` kí tự: kí tự đầu tiên thể hiện cột, kí tự thứ hai thể hiện hàng. Ví dụ như `a8, b3, c2, ...`



Cho biết vị trí của con mã trên bàn cờ vua, hãy tìm số vị trí khác nhau mà con mã có thể thực hiện nước nhảy

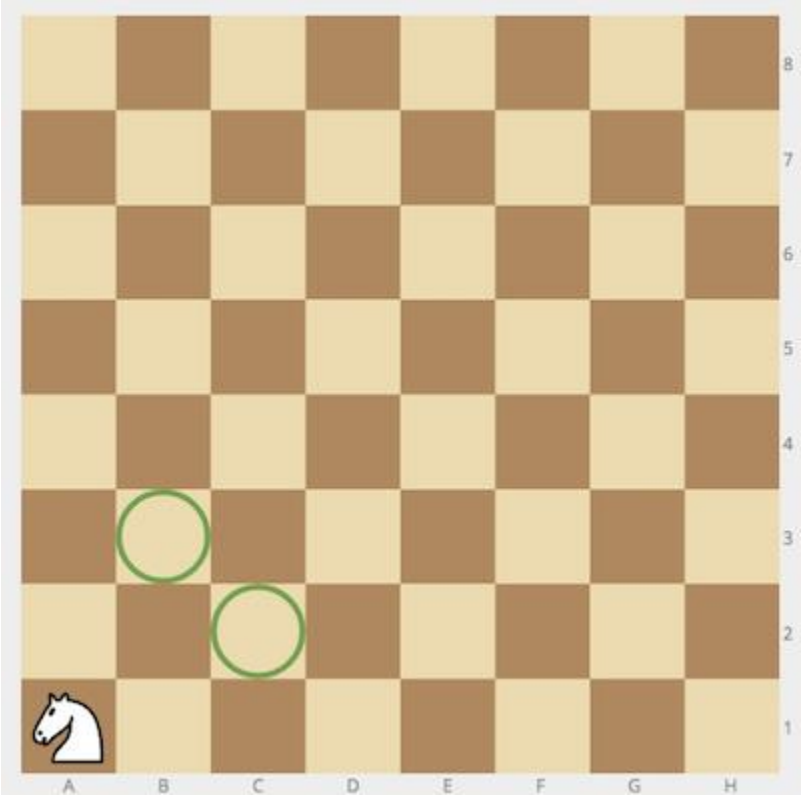
Một con mã có thể nhảy tới 1 vị trí cách vị trí hiện tại `2` ô theo chiều ngang và `1` ô theo chiều dọc, hoặc `2` ô theo chiều dọc và `1` ô theo chiều ngang. Nước đi của con mã tạo ra hình dáng giống như `1` chữ `L`

Xem hình bên dưới để thấy rõ các nước đi hợp lệ của một con mã

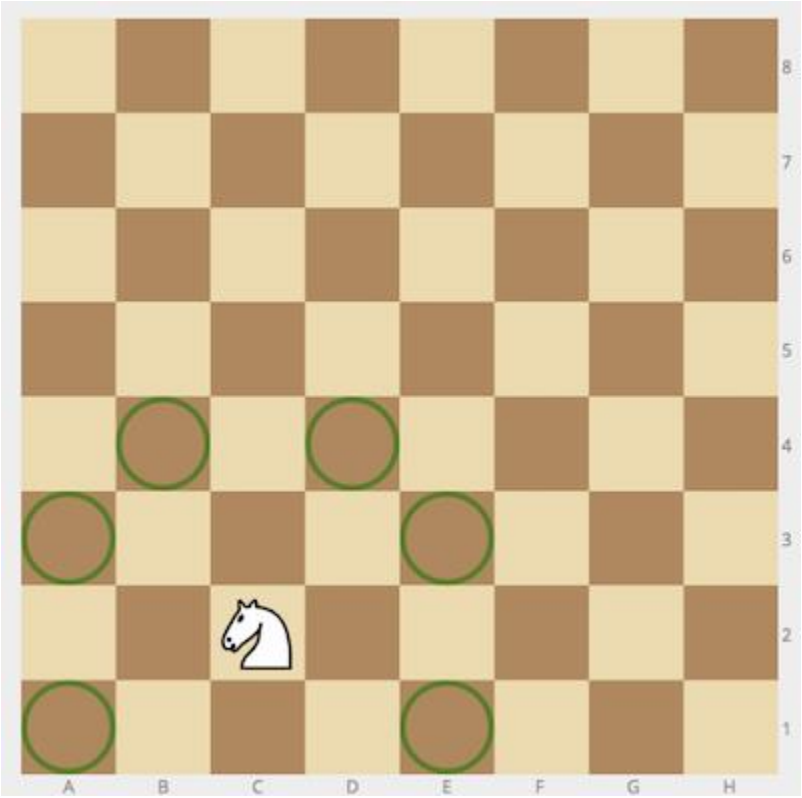


Ví dụ:

- Với `cell = "a1"`, thì kết quả `chessKnight(cell) = 2`.



- Với `cell = "c2"`, thì kết quả `chessKnight(cell) = 6`.



Đầu vào/Đầu ra

- **[Thời gian chạy] 0.5 giây**
- **[Đầu vào] string** cell
  Vị trí hiện tại của con mã
  Điều kiện:
  `cell.length = 2`,
  `'a' ≤ cell[0] ≤ 'h'`,
  `1 ≤ cell[1] ≤ 8`.
- **[Đầu ra] integer**
  Số nước đi mà con mã có thể thực hiện

Lý thuyết :

- Cách để duyệt các tọa độ sắp truy cập từ ô hiện tại (được áp dụng rất nhiều sau này, khi các bạn làm đến những bài có BFS trên bảng) : Dùng 2 bảng lưu độ chênh về giá trị hàng, cột của ô sắp đến so với ô hiện tại. Rồi for và duyệt, xử lí chung với các ô tọa độ mới thay vì xử lí tay với từng ô
- Cụ thể hơn, để dễ hình dung, mình sẽ đưa ra 1 bài ví dụ sau : Ô hiện tại là ô `(x, y)`. Kiểm tra xem có bao nhiêu ô kề cạnh với ô `(x, y)` nằm trong bảng `[1 ... n, 1 ... m]`
- Thuật toán : Bạn sẽ muốn từ ô hiện tại thăm 4 ô kề cạnh xung quanh, kiểm tra xem chúng có ở trong bảng không
- **Cách 1** : bạn sẽ viết tay tọa độ 4 ô, xử lí chúng lần lượt

```
int CountInBoard(int x, int y, int m, int n) {
	int res = 0; // ket qua tra ve so o xung quanh o (x, y) trong bang [1 .. m, 1 .. n]
	if (x - 1 > 0 && y > 0 && x - 1 <= m && y <= n) ++res; // kiem tra voi o (x - 1, y)
	if (x > 0 && y - 1 > 0 && x <= m && y - 1 <= n) ++res; // kiem tra voi o (x, y - 1)
	if (x > 0 && y + 1 > 0 && x <= m && y + 1 <= n) ++res; // kiem tra voi o (x, y + 1)
	if (x + 1 > 0 && y > 0 && x + 1 <= m && y <= n) ++res; // kiem tra voi o (x + 1, y)
	return res;
}
```

- **Cách 2** : bạn dùng mảng lưu độ chênh lệch rồi duyệt

```
int row[] = { -1, 0, 1, 0 }; // chenh lech ve hang
int col[] = { 0, 1, 0, -1 }; // chenh lech ve cot
int CountInBoard(int x, int y, int m, int n) {
	int res = 0; // ket qua tra ve
	for (int dir = 0; dir < 4; ++dir) {
		int nx = x + row[dir], ny = y + col[dir]; // toa do cua o xung quanh
		if (nx > 0 && ny > 0 && nx <= m && ny <= n) ++res;
			// kiem tra xem o nam trong bang khong
	}
	return res;
```