**★★★☆☆**

**題組：Problem Set Archive** **with Online Judge**

**題號：439: Knight Moves**

**解題者：鍾承恩**

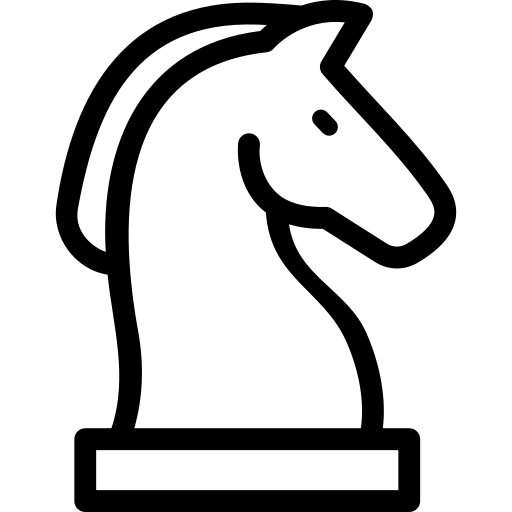
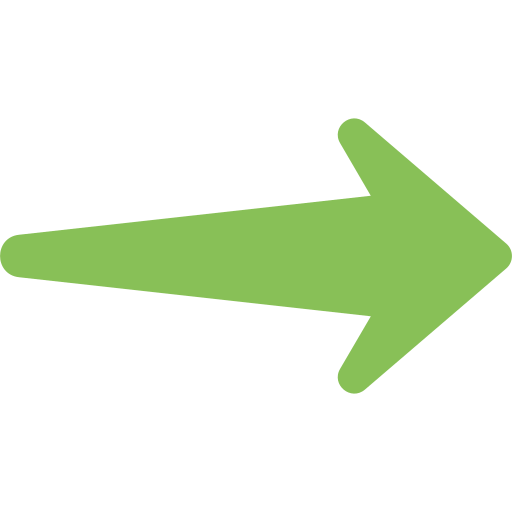
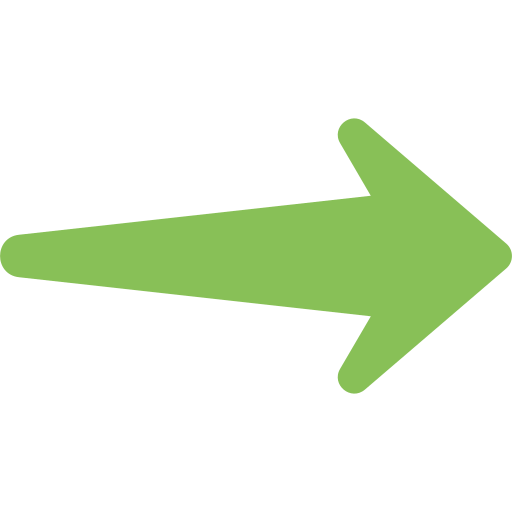
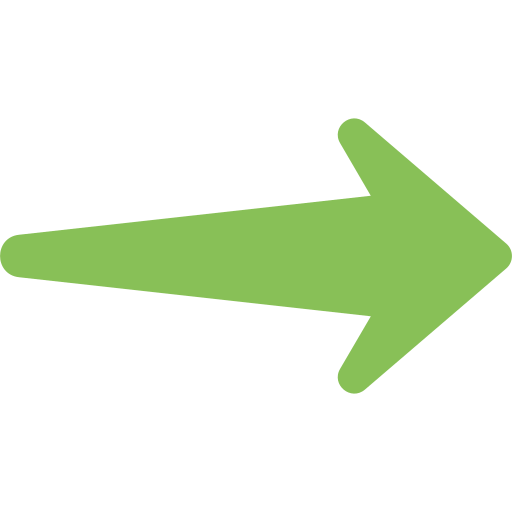
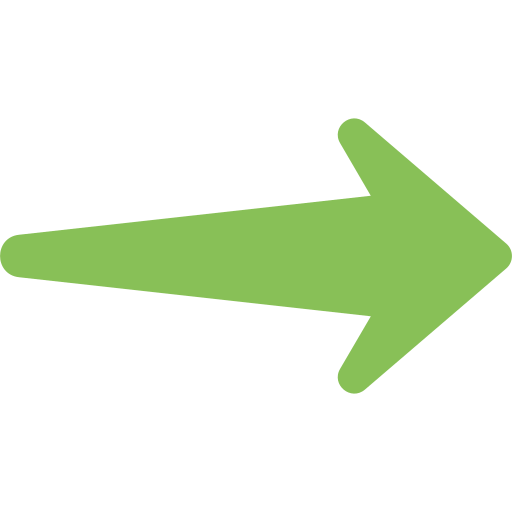
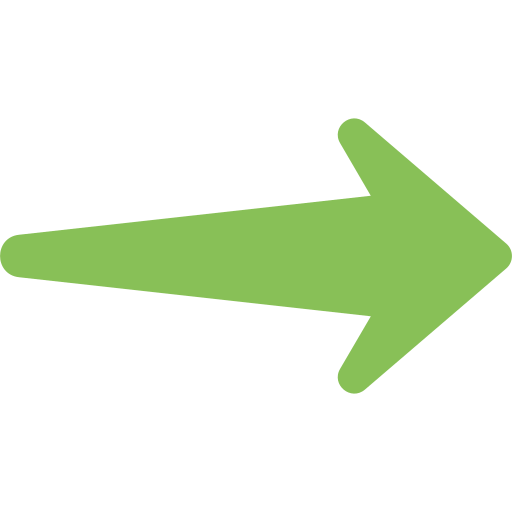
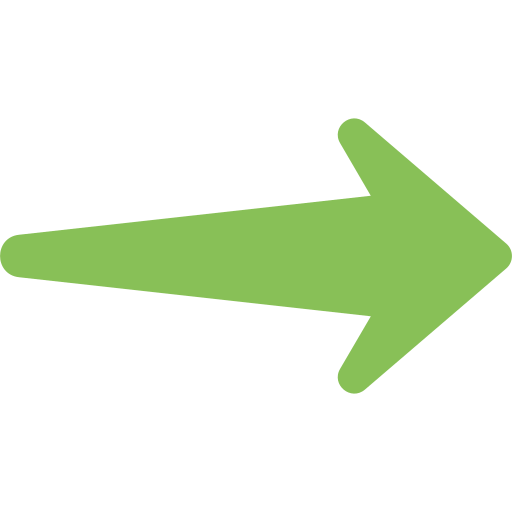
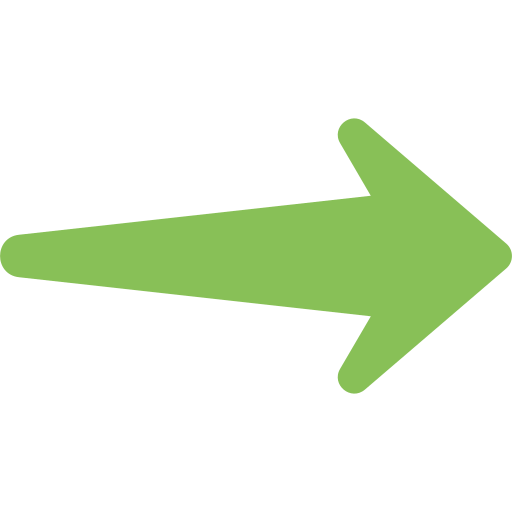
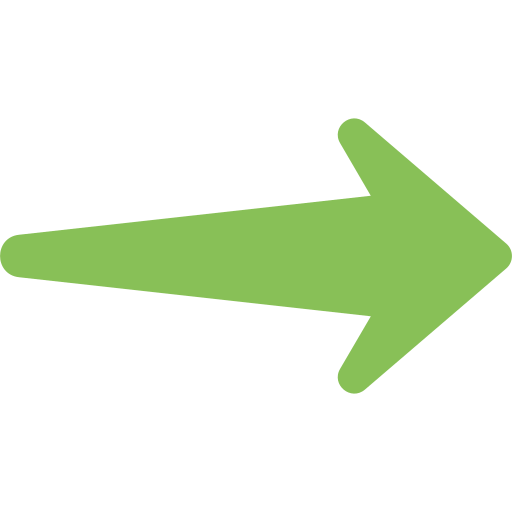
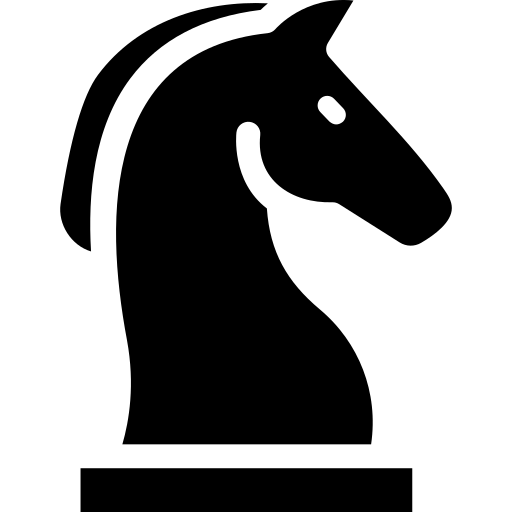
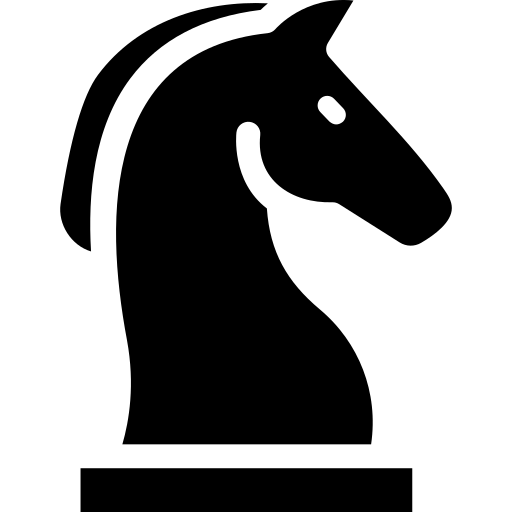
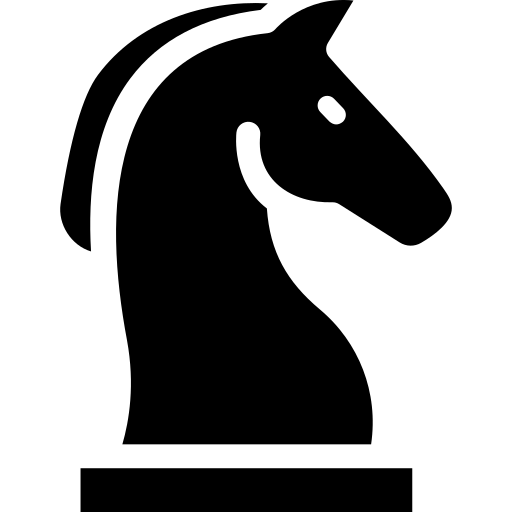
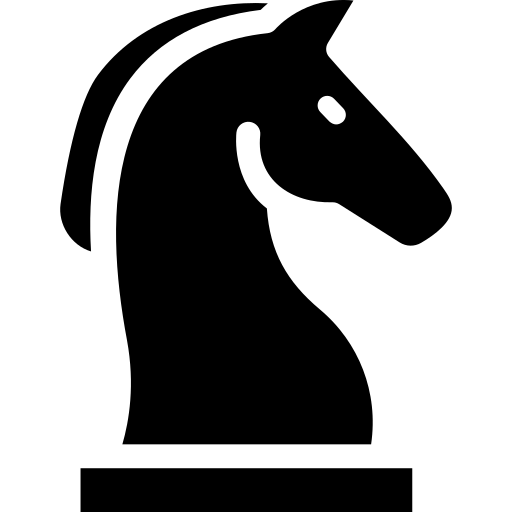
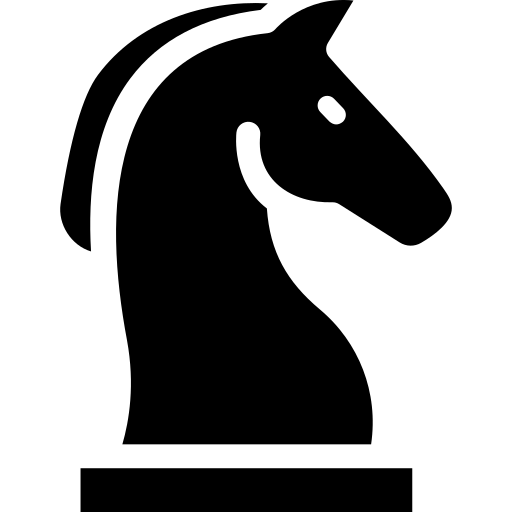
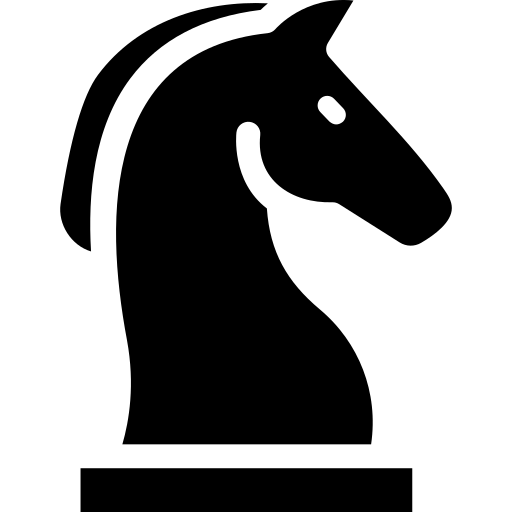
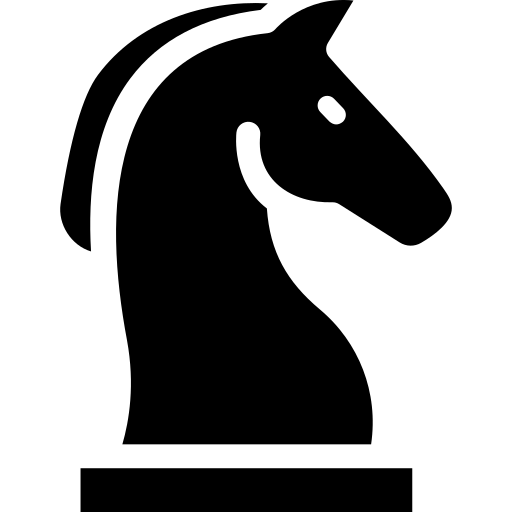
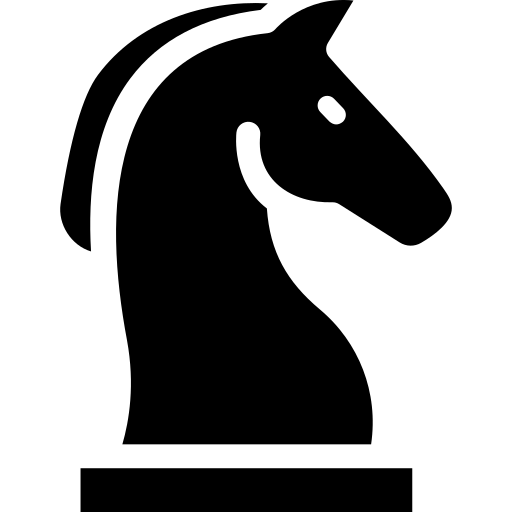
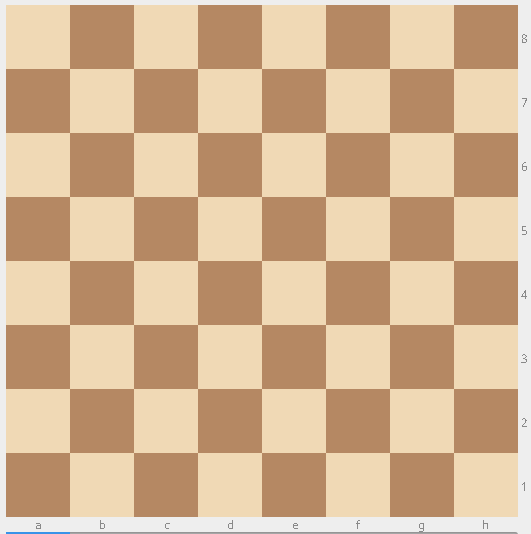
**解題日期：2024年5月2日**

**題意：**

騎士只能在一個8 × 8大小的棋盤上行動，其中每一行依序由a ~ h標示，每一列依序由1 ~ 8標示。給訂一個起點位置及終點位置 ，輸出騎士從起點到終點所需的最少步數: To get from xx to yy takes n knight moves。

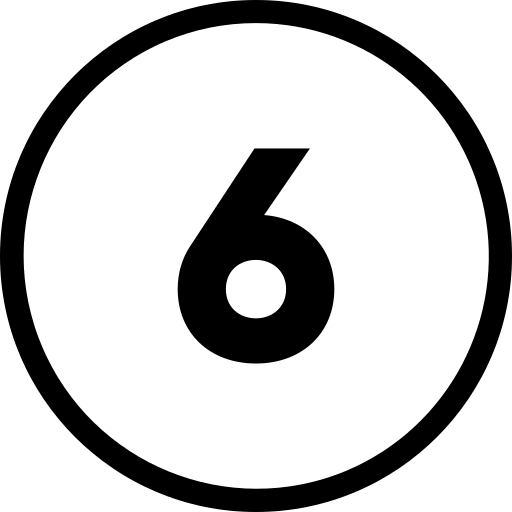
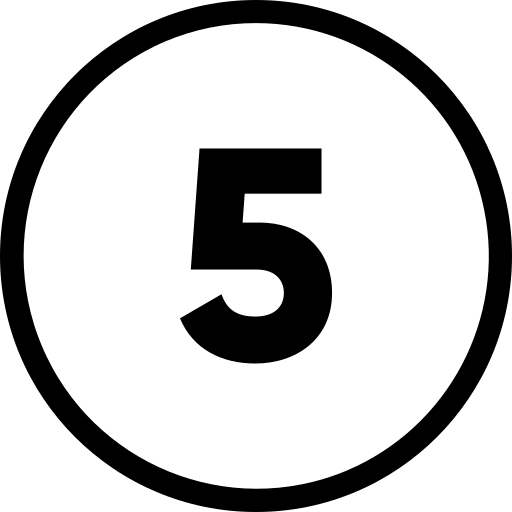
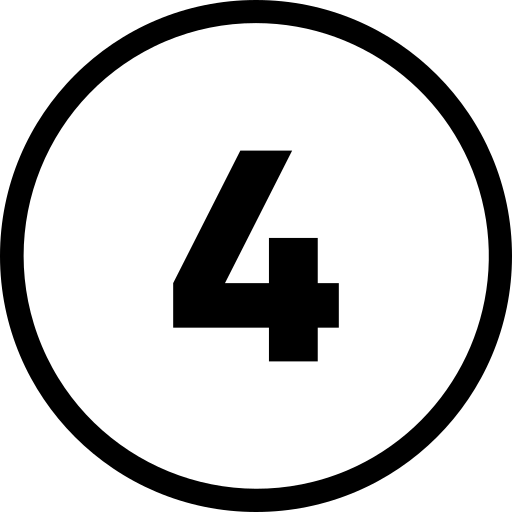
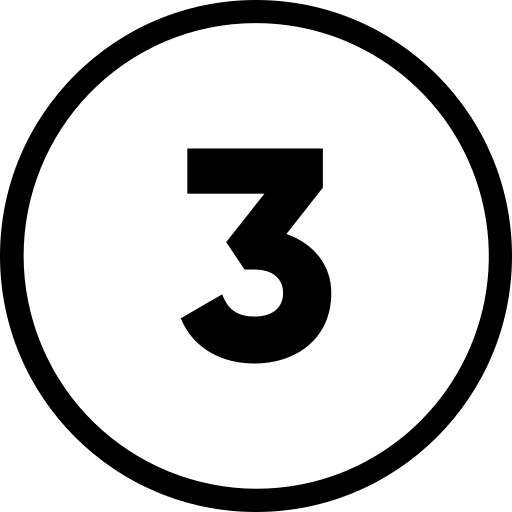
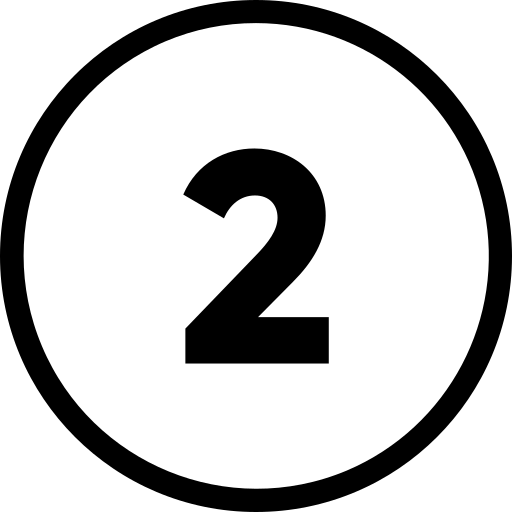
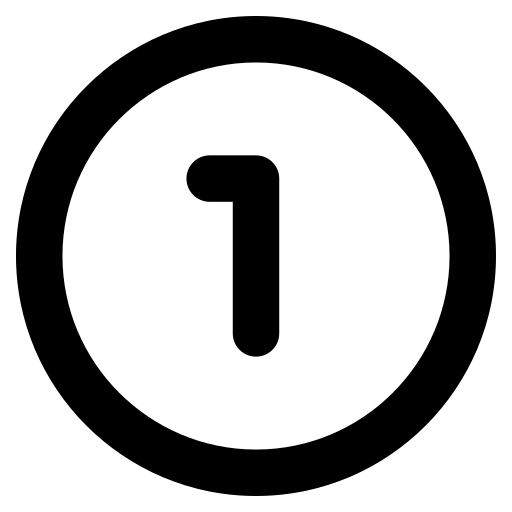
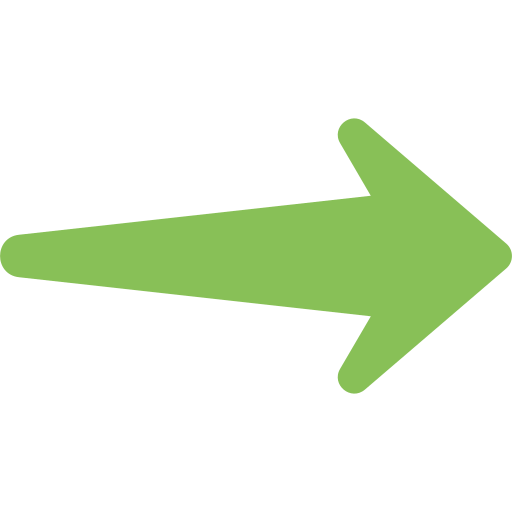
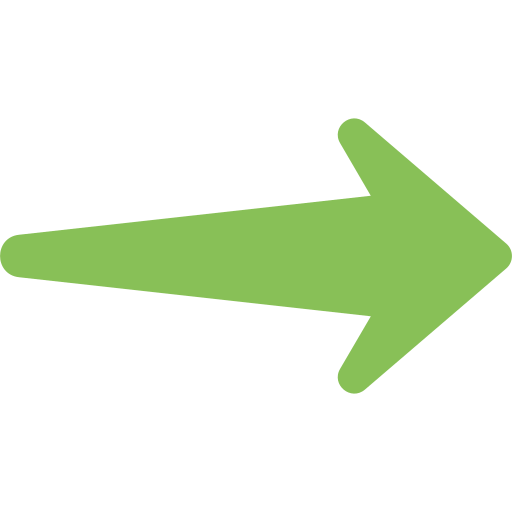
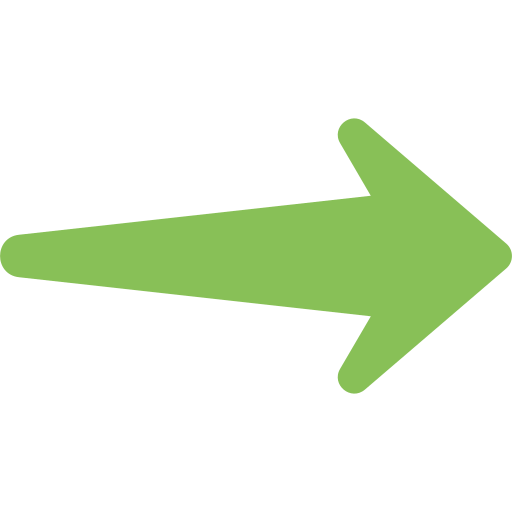
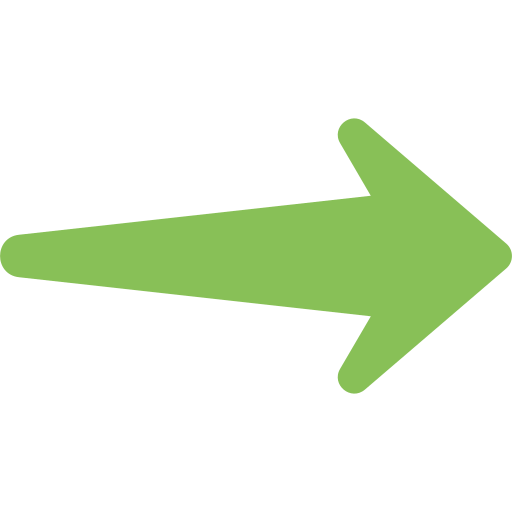
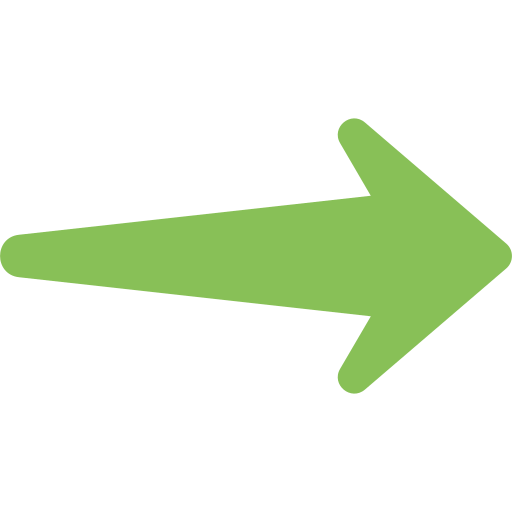
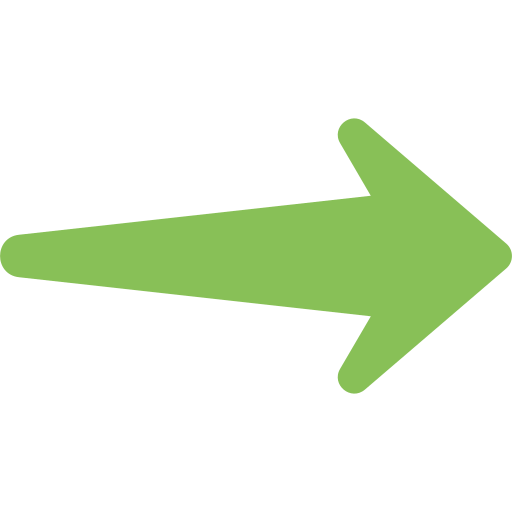
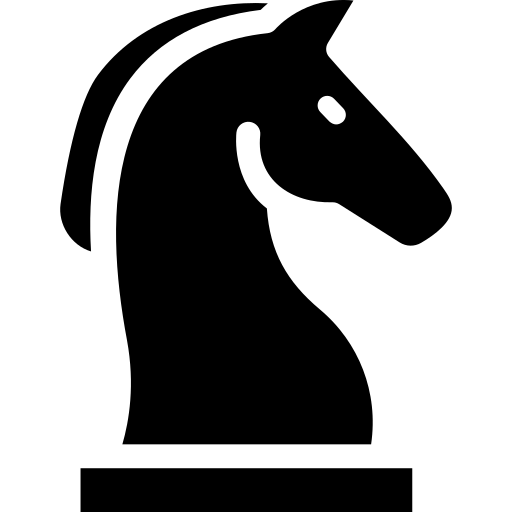
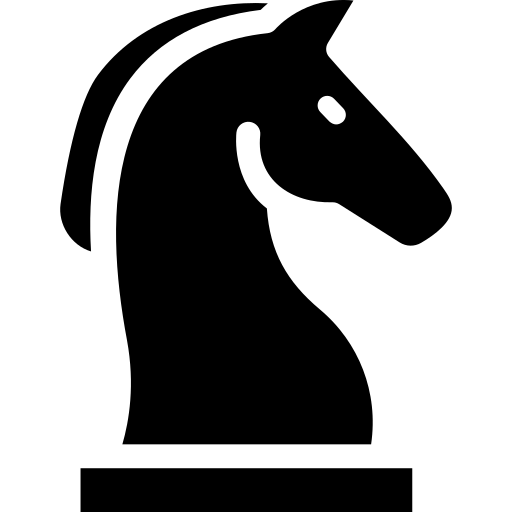
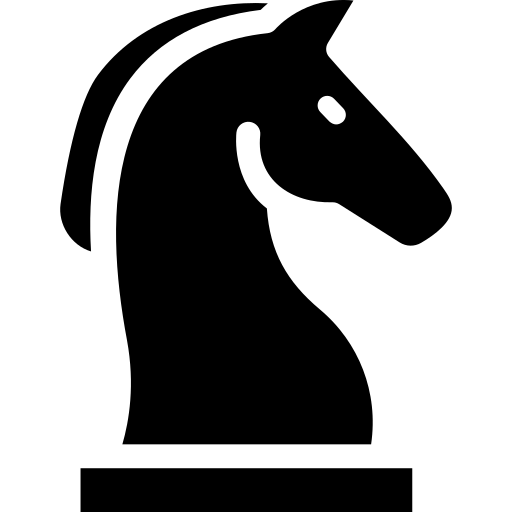
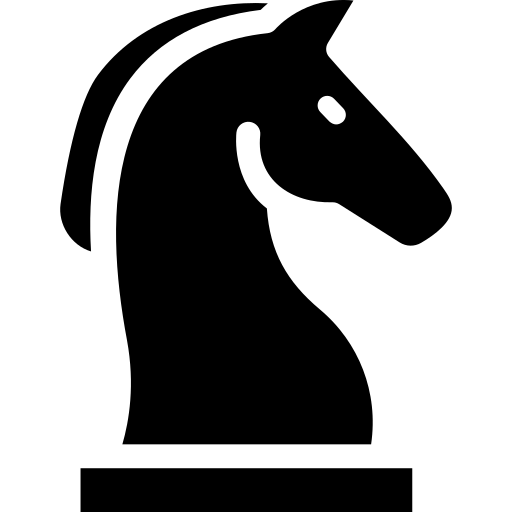
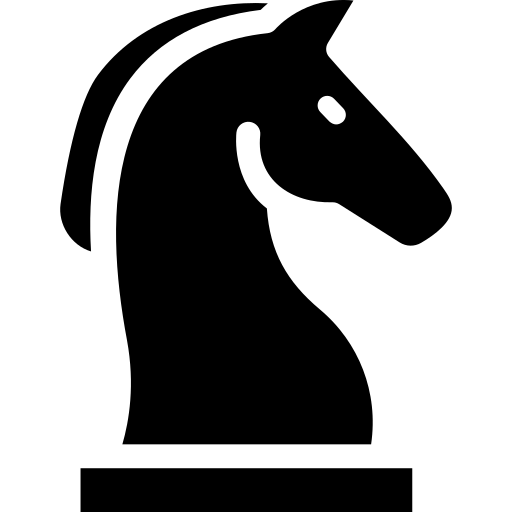
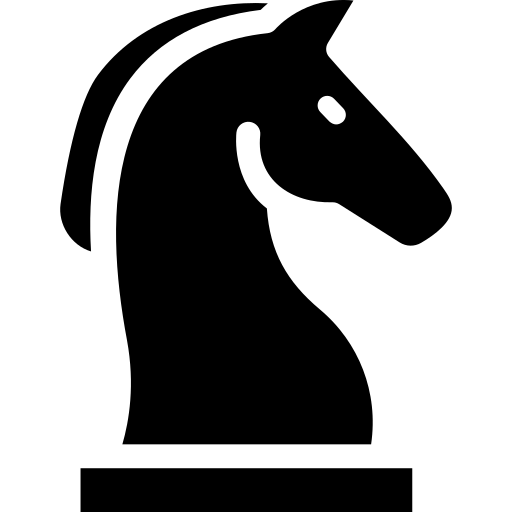
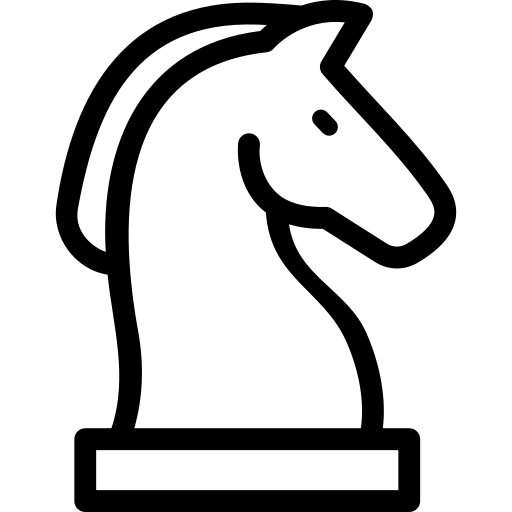
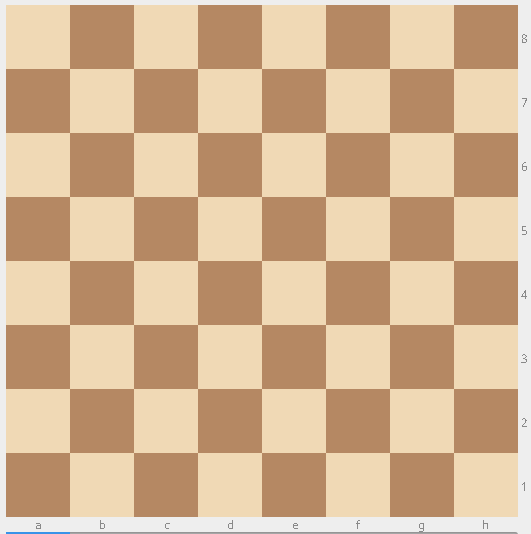
e2 e4 // 從e2走到e4  
 a1 b2 // 從a1走到b2

**題意範例：**



e2 e4

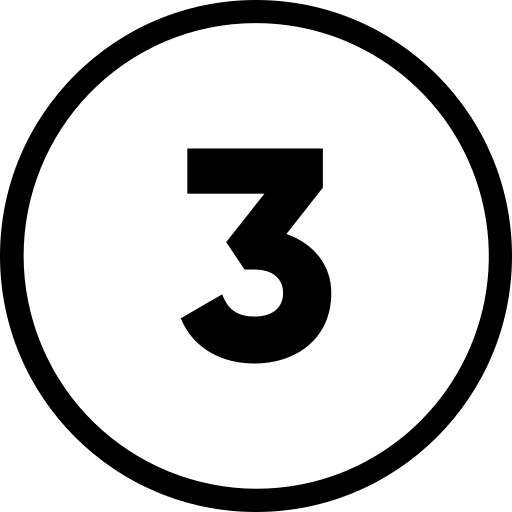
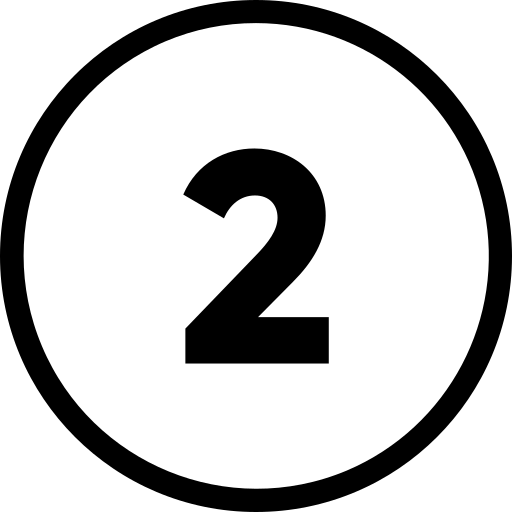
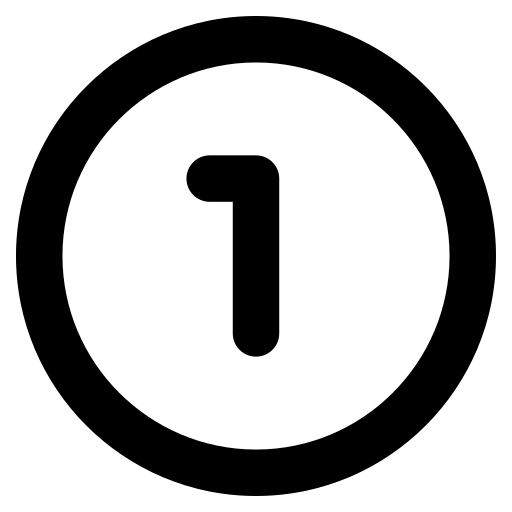
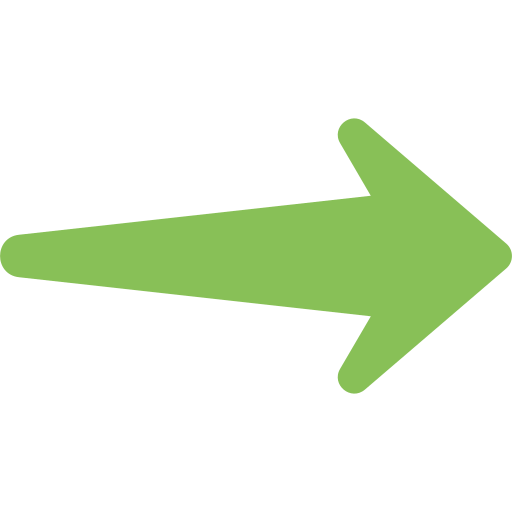
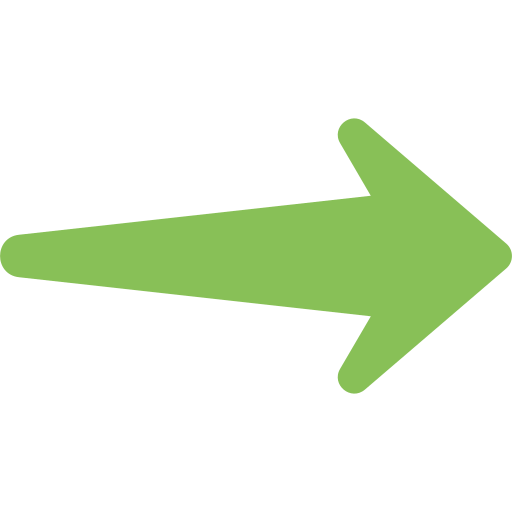
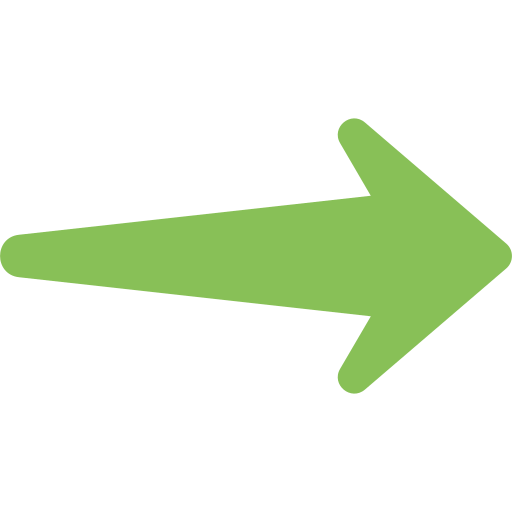
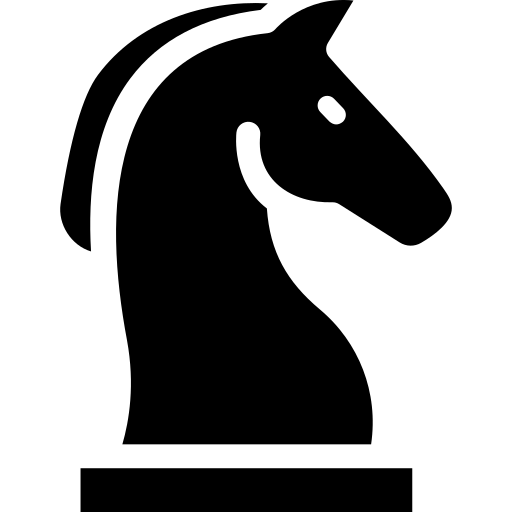
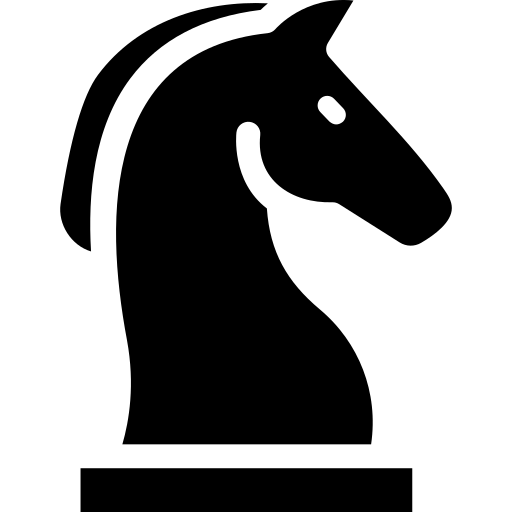
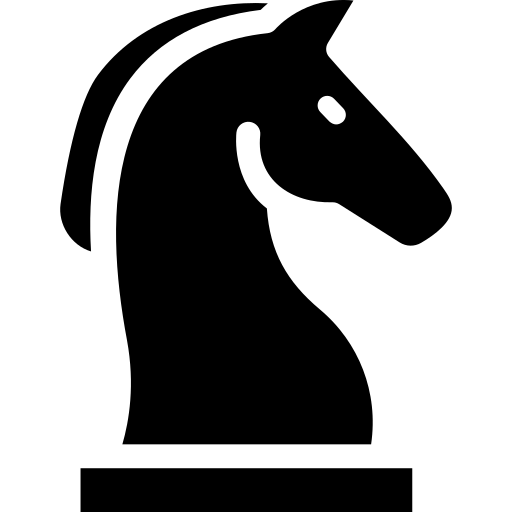
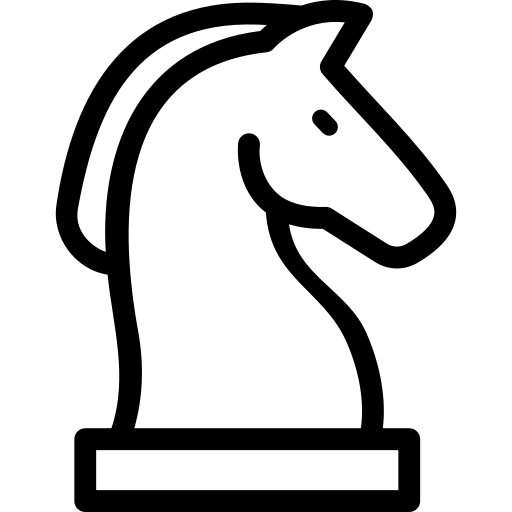
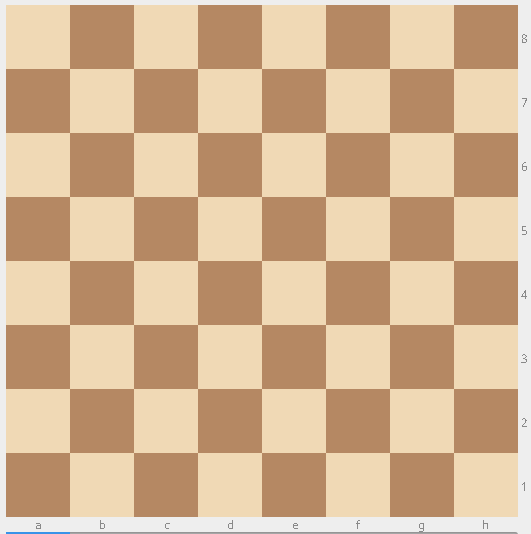
To get from e2 to e4 takes 2 knight moves.



1st move

e2 e4

To get from e2 to e4 takes 2 knight moves.



2nd move

**解法：**

利用BFS將騎士當前可走的每個位置都走過一遍，紀錄當前步數並且紀錄該位置已走訪，執行直到第一次走到終點位置則輸出，若無其他測資則程式結束。

**解法範例：**

無

**討論：**

(1) 令n為棋盤行數及列數，騎士在任一位置可行動的下一步最 多為8種，則時間複雜度為 O(8 x n x n) = O(n^2)。  
(2) 每次走到新的座標時，紀錄當前位置已走訪，並且將該座標新增到隊列中。

**程式：**

#include <iostream>

using namespace std;

int start\_x, start\_y, end\_x, end\_y, cur\_x, cur\_y, temp\_x, temp\_y, ans;

int queue[65][2], dist[8][8];

int dx[8] = {1,1,-1,-1,2,2,-2,-2};

int dy[8] = {2,-2,2,-2,1,-1,1,-1};

char SX,SY,DX,DY;

bool vis[8][8];

int BFS(){

dist[start\_x][start\_y] = 0;

queue[0][0] = start\_x;

queue[0][1] = start\_y;

vis[start\_x][start\_y] = true;

for(int i=1,j=0;j<i;j++){

cur\_x = queue[j][0];

cur\_y = queue[j][1];

if(cur\_x==end\_x && cur\_y==end\_y){

// find the position;

return dist[cur\_x][cur\_y];

}

for(int move=0;move<8;move++){

temp\_x = cur\_x+dx[move];

temp\_y = cur\_y+dy[move];

if(temp\_x<0 || temp\_x>7 || temp\_y<0 || temp\_y>7){

continue;

}

if(!vis[temp\_x][temp\_y]){

vis[temp\_x][temp\_y] = true;

dist[temp\_x][temp\_y] = dist[cur\_x][cur\_y]+1;

queue[i][0] = temp\_x;

queue[i][1] = temp\_y;

i++;

}

}

}

return -1;

}

int main(){

while(cin >> SX >> SY >> DX >> DY){

start\_x = SX-'a';

start\_y = SY-'1';

end\_x = DX-'a';

end\_y = DY-'1';

for(int i=0;i<8;i++){

for(int j=0;j<8;j++){

dist[i][j] = 0;

vis[i][j] = false;

}

}

ans = BFS();

cout << "To get from " << SX << SY << " to " << DX << DY << " takes " << ans << " knight moves.\n";

}

}