|  |
| --- |
| #include<iostream> |
|  | #include<queue> |
|  | #define INT\_MAX 1000000 |
|  | using namespace std; |
|  |  |
|  |  |
|  |  |
|  | int findMinMoves(int \*board,int n){ |
|  | //We will use BFS to calculate shortest path |
|  | //moves[i] maintains the min number of moves to reach position i |
|  | int moves[n+6]; |
|  | for(int i=0;i<n;i++){ |
|  | moves[i] = INT\_MAX; |
|  | } |
|  | queue<int> q; |
|  | /\* Game starts from 0 \*/ |
|  | q.push(0); |
|  | moves[0] = 0; |
|  |  |
|  | int parent[37]; |
|  | parent[0]=0; |
|  |  |
|  | while(!q.empty()){ |
|  | int current = q.front(); |
|  | q.pop(); |
|  | for(int i=1;i<=6;i++){ |
|  | if(current+i<n) |
|  | { |
|  | if(moves[current+i+board[current+i]]==INT\_MAX){ |
|  | /\* As BFS calculates shortest path on first time,so we need to update move matrix only once \*/ |
|  | moves[current+i+board[current+i]] = moves[current]+1; |
|  | q.push(current+i+board[current+i]); |
|  |  |
|  | /\* Parent is required to print the path \*/ |
|  | parent[current+i+board[current+i]] = current; |
|  | } |
|  | } |
|  | } |
|  | } |
|  | /\* Code to print the moves \*/ |
|  | int i=n-1; |
|  | cout<<n-1<<"<--"; |
|  | while(i!=0){ |
|  | cout<<parent[i]<<" <--"; |
|  | i = parent[i]; |
|  | } |
|  | cout<<endl; |
|  |  |
|  | /\*Return the number of Moves \*/ |
|  | return moves[n-1]; |
|  | } |
|  |  |
|  |  |
|  | int main(){ |
|  | /\*Since it is a linear games (1,2....n) we need to maintain only a linear array for board |
|  | Note : We don't require a 2-D array. |
|  | \*/ |
|  |  |
|  | int board[37] ={0}; //Game Index is 1 based |
|  | /\*All board positions are 0 except those which have ladder have +ve values , Snakes have |
|  | -ve values .Values are assigned on the basis of relative displacement between two positions.\*/ |
|  | board[2] = 13; |
|  | board[5] = 2; |
|  | board[9] = 28; |
|  | board[18] = 11; |
|  | board[17] = -13; |
|  | board[20] = -14; |
|  | board[24] = -8; |
|  | board[25] = 10; |
|  | board[32] = -2; |
|  | board[34] = -22; |
|  | /\* |
|  | for(int i=0;i<36;i++) |
|  | cout<<board[i]<<" "; |
|  | \*/ |
|  | cout<<"The min no of moves required to reach 36 is "<<findMinMoves(board,37); |
|  | cout<<endl; |
|  | return 0; |
|  | } |