#include<bits/stdc++.h>

using namespace std;

// sr - source row

// sc - source column

// dr - destination row

// dc - destination column

vector<string> solve(int sr, int sc, int dr, int dc){

if(sr==dr && sc==dc){

vector<string> base;

base.push\_back("");

return base;

}

vector<string> res;

vector<string> hpaths;

vector<string> vpaths;

vector<string> dpaths;

int i=1,j=1,k=1;

while(i<=dc-sc){

hpaths = solve(sr,sc+i,dr,dc);

string ii = to\_string(i);

for(auto paths: hpaths){

string s = "h"+ii+paths;

res.push\_back(s);

}

i++;

}

while(j<=dr-sr){

vpaths = solve(sr+j,sc,dr,dc);

string ii = to\_string(j);

for(auto paths: vpaths){

string s = "v"+ii+paths;

res.push\_back(s);

}

j++;

}

while(k<=dc-sc && k<=dr-sr){

dpaths = solve(sr+k,sc+k,dr,dc);

string ii = to\_string(k);

for(auto paths: dpaths){

string s = "d"+ii+paths;

res.push\_back(s);

}

k++;

}

return res;

}

void display(vector<string>& arr){

cout << "[";

for(int i = 0;i < arr.size();i++){

cout << arr[i];

if(i < arr.size() -1) cout << ", ";

}

cout << "]"<<endl;

}

int main() {

int n,m; cin >> n >> m;

// vector<string> ans = getMazePaths(n,m);

vector<string> ans = solve(1,1,n,m);

display(ans);

return 0;

}