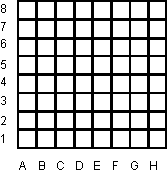
**Camelot**  
**IOI 98**

Centuries ago, King Arthur and the Knights of the Round Table used to meet every year on New Year's Day to celebrate their fellowship. In remembrance of these events, we consider a board game for one player, on which one chesspiece king and several knight pieces are placed on squares, no two knights on the same square.

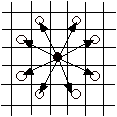
This example board is the standard 8x8 array of squares:



The King can move to any adjacent square from http://ace.delos.com/usaco/probs/camelot-b.gifto http://ace.delos.com/usaco/probs/camelot-w.gifas long as it does not fall off the board:



A Knight can jump from http://ace.delos.com/usaco/probs/camelot-b.gifto http://ace.delos.com/usaco/probs/camelot-w.gif, as long as it does not fall off the board:



During the play, the player can place more than one piece in the same square. The board squares are assumed big enough so that a piece is never an obstacle for any other piece to move freely.

The player's goal is to move the pieces so as to gather them all in the same square - in the minimal number of moves. To achieve this, he must move the pieces as prescribed above. Additionally, whenever the king and one or more knights are placed in the same square, the player may choose to move the king and one of the knights together from that point on, as a single knight, up to the final gathering point. Moving the knight together with the king counts as a single move.

Write a program to compute the minimum number of moves the player must perform to produce the gathering. The pieces can gather on any square, of course.

**PROGRAM NAME: camelot**

**INPUT FORMAT**

|  |  |
| --- | --- |
| Line 1: | Two space-separated integers: R,C, the number of rows and columns on the board. There will be no more than 26 columns and no more than 30 rows. |
| Line 2..end: | The input file contains a sequence of space-separated letter/digit pairs, 1 or more per line. The first pair represents the board position of the king; subsequent pairs represent positions of knights. There might be 0 knights or the knights might fill the board. Rows are numbered starting at 1; columns are specified as upper case characters starting with `A'. |

**SAMPLE INPUT (file camelot.in)**

8 8

D 4

A 3 A 8

H 1 H 8

The king is positioned at D4. There are four knights, positioned at A3, A8, H1, and H8.

**OUTPUT FORMAT**

A single line with the number of moves to aggregate the pieces.

**SAMPLE OUTPUT (file camelot.out)**

10

**SAMPLE OUTPUT ELABORATION**

They gather at B5.   
Knight 1: A3 - B5 (1 move)   
Knight 2: A8 - C7 - B5 (2 moves)   
Knight 3: H1 - G3 - F5 - D4 (picking up king) - B5 (4 moves)   
Knight 4: H8 - F7 - D6 - B5 (3 moves)   
1 + 2 + 4 + 3 = 10 moves.

http://ace.delos.com/clear_dot.gif

USER: Neal Gavin Gavin [nealgav1]

TASK: camelot

LANG: C++

Compiling...

Compile: OK

Executing...

Test 1: TEST OK [0.000 secs, 3380 KB]

Test 2: TEST OK [0.000 secs, 3380 KB]

Test 3: TEST OK [0.000 secs, 3380 KB]

Test 4: TEST OK [0.022 secs, 3380 KB]

Test 5: TEST OK [0.151 secs, 3380 KB]

Test 6: TEST OK [0.238 secs, 3380 KB]

Test 7: TEST OK [0.000 secs, 3380 KB]

Test 8: TEST OK [0.000 secs, 3380 KB]

Test 9: TEST OK [0.086 secs, 3380 KB]

Test 10: TEST OK [0.389 secs, 3380 KB]

Test 11: TEST OK [0.000 secs, 3380 KB]

Test 12: TEST OK [0.000 secs, 3380 KB]

Test 13: TEST OK [0.000 secs, 3380 KB]

Test 14: TEST OK [0.000 secs, 3380 KB]

Test 15: TEST OK [0.000 secs, 3380 KB]

Test 16: TEST OK [0.000 secs, 3380 KB]

Test 17: TEST OK [0.000 secs, 3380 KB]

Test 18: TEST OK [0.000 secs, 3380 KB]

Test 19: TEST OK [0.000 secs, 3380 KB]

Test 20: TEST OK [0.000 secs, 3380 KB]

All tests OK.

Your program ('camelot') produced all correct answers! This is your

submission #3 for this problem. **Congratulations!**

**思路：****先求出王到所有点的最少时间，骑士到每个点的最少时间，然后可以骑士带着王走，骑士的时间假设其不变，只更新王所到点的时间。最后枚举所有点加和最小就是答案。**

/\*

ID:nealgav1

LANG:C++

PROG:camelot

\*/

#include<fstream>

#include<cstring>

#include<queue>

using namespace std;

ifstream cin("camelot.in");

ofstream cout("camelot.out");

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

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

const int oo=1e9;

class node {

public:

int x, y;

void data(int a,int b){x=a,y=b;}

};

int R, C;

int cost[26][30];

int kcost[26][30];

int dist[26][30];

int kdist[26][30];

int kkdist[26][30]; // original king steps

bool vis[26][30];

bool isKnight[26][30];

void bfs(int sx, int sy)

{

//int steps = 0;

int x, y;

node p;

memset(vis, false, 30\*26);

memset(dist, 0, sizeof(int)\*30\*26);

queue<node> Q;

p.data(sx,sy);

Q.push(p);

//dist[sx][sy] = 1;

vis[sx][sy] = true;

//bfs alone

while (!Q.empty()) {

p = Q.front(); Q.pop();

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

x = p.x+dx[i];

y = p.y+dy[i];

// check if the next node is vaild

if (x>=0 && x<C && y>=0 && y<R && !vis[x][y]) {

vis[x][y] = true;

node s;s.data(x,y);

Q.push(s);

dist[x][y] = dist[p.x][p.y] + 1; // one move

cost[x][y] += dist[x][y]; // add a knight

}

}

}

for (int i = 0; i < C; i++)

for (int j = 0; j < R; j++)

kdist[i][j] = kkdist[i][j];

// bfs with king

for (int i = 0; i < C; i++)

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

node s;s.data(i,j);

Q.push(s);

while (!Q.empty()) {

p = Q.front(); Q.pop();

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

x = p.x + dx[k];

y = p.y + dy[k];

if (x>=0 && x<C && y>=0 && y<R) {

if (kdist[x][y] + dist[x][y] > dist[p.x][p.y] + kdist[p.x][p.y] + 1) {

kdist[x][y] = dist[p.x][p.y] + kdist[p.x][p.y] + 1 - dist[x][y];

s.data(x,y);

Q.push(s);

}

}

}

}

}

for (int i = 0; i < C; i++)

for (int j = 0; j < R; j++)

if (kdist[i][j] < kcost[i][j])

kcost[i][j] = kdist[i][j];

}

int \_abs(int x)

{

return x>0?x:-x;

}

int main()

{

char xx;

int x, y;

while(cin>>R>>C)

{

cin >> xx >> y;

x = xx - 'A';

y --;

for (int i = 0; i < C; i++)

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

if (\_abs(i-x) > \_abs(j-y))

kcost[i][j] = kkdist[i][j] = \_abs(i-x);

else

kcost[i][j] = kkdist[i][j] = \_abs(j-y);

}

while (cin >> xx >> y) {

x = xx - 'A';

y --;

isKnight[x][y] = true;

bfs(x, y); // bfs

}

int ans =oo;

for (int i = 0; i < C; i++)

for (int j = 0; j < R; j++)

if ((cost[i][j] > 0 || isKnight[i][j]) && ans > cost[i][j] + kcost[i][j]) {

ans = cost[i][j] + kcost[i][j];

}

if (ans == oo) //no knight

cout<<"0\n";

else {

cout << ans << endl;

}

}

return 0;

}

Here are the test data inputs:

------- test 1 ----

1 1

A 1

------- test 2 ----

8 8

A 1

C 2 A 1

------- test 3 ----

8 8

A 1

A 1 A 2 A 3 A 4 A 5 A 6 A 7 A 8 B 1 B 2 B 3 B 4 B 5 B 6 B 7 B 8 C 1 C 2 C 3 C 4 C 5

C 6 C 7 C 8 D 1 D 2 D 3 D 4 D 5 D 6 D 7 D 8 E 1 E 2 E 3 E 4 E 5 E 6 E 7 E 8 F 1 F 2

F 3 F 4 F 5 F 6 F 7 F 8 G 1 G 2 G 3 G 4 G 5 G 6 G 7 G 8 H 1 H 2 H 3 H 4 H 5 H 6 H 7

H 8

------- test 4 ----

15 15

A 1

A 1 A 2 A 3 A 4 A 5 A 6 A 7 A 8 A 9 A 10 A 11 A 12 A 13 A 14 A 15 B 1 B 2 B 3 B 4 B 5 B 6

B 7 B 8 B 9 B 10 B 11 B 12 B 13 B 14 B 15 C 1 C 2 C 3 C 4 C 5 C 6 C 7 C 8 C 9 C 10 C 11 C 12

C 13 C 14 C 15 D 1 D 2 D 3 D 4 D 5 D 6 D 7 D 8 D 9 D 10 D 11 D 12 D 13 D 14 D 15 E 1 E 2 E 3

E 4 E 5 E 6 E 7 E 8 E 9 E 10 E 11 E 12 E 13 E 14 E 15 F 1 F 2 F 3 F 4 F 5 F 6 F 7 F 8 F 9

F 10 F 11 F 12 F 13 F 14 F 15 G 1 G 2 G 3 G 4 G 5 G 6 G 7 G 8 G 9 G 10 G 11 G 12 G 13 G 14 G 15

H 1 H 2 H 3 H 4 H 5 H 6 H 7 H 8 H 9 H 10 H 11 H 12 H 13 H 14 H 15 I 1 I 2 I 3 I 4 I 5 I 6

I 7 I 8 I 9 I 10 I 11 I 12 I 13 I 14 I 15 J 1 J 2 J 3 J 4 J 5 J 6 J 7 J 8 J 9 J 10 J 11 J 12

J 13 J 14 J 15 K 1 K 2 K 3 K 4 K 5 K 6 K 7 K 8 K 9 K 10 K 11 K 12 K 13 K 14 K 15 L 1 L 2 L 3

L 4 L 5 L 6 L 7 L 8 L 9 L 10 L 11 L 12 L 13 L 14 L 15 M 1 M 2 M 3 M 4 M 5 M 6 M 7 M 8 M 9

M 10 M 11 M 12 M 13 M 14 M 15 N 1 N 2 N 3 N 4 N 5 N 6 N 7 N 8 N 9 N 10 N 11 N 12 N 13 N 14 N 15

O 1 O 2 O 3 O 4 O 5 O 6 O 7 O 8 O 9 O 10 O 11 O 12 O 13 O 14 O 15

------- test 5 ----

25 25

A 1

A 1 A 2 A 3 A 4 A 5 A 6 A 7 A 8 A 9 A 10 A 11 A 12 A 13 A 14 A 15 A 16 A 17 A 18 A 19 A 20 A 21

A 22 A 23 A 24 A 25 B 1 B 2 B 3 B 4 B 5 B 6 B 7 B 8 B 9 B 10 B 11 B 12 B 13 B 14 B 15 B 16 B 17

B 18 B 19 B 20 B 21 B 22 B 23 B 24 B 25 C 1 C 2 C 3 C 4 C 5 C 6 C 7 C 8 C 9 C 10 C 11 C 12 C 13

C 14 C 15 C 16 C 17 C 18 C 19 C 20 C 21 C 22 C 23 C 24 C 25 D 1 D 2 D 3 D 4 D 5 D 6 D 7 D 8 D 9

D 10 D 11 D 12 D 13 D 14 D 15 D 16 D 17 D 18 D 19 D 20 D 21 D 22 D 23 D 24 D 25 E 1 E 2 E 3 E 4 E 5

E 6 E 7 E 8 E 9 E 10 E 11 E 12 E 13 E 14 E 15 E 16 E 17 E 18 E 19 E 20 E 21 E 22 E 23 E 24 E 25 F 1

F 2 F 3 F 4 F 5 F 6 F 7 F 8 F 9 F 10 F 11 F 12 F 13 F 14 F 15 F 16 F 17 F 18 F 19 F 20 F 21 F 22

F 23 F 24 F 25 G 1 G 2 G 3 G 4 G 5 G 6 G 7 G 8 G 9 G 10 G 11 G 12 G 13 G 14 G 15 G 16 G 17 G 18

G 19 G 20 G 21 G 22 G 23 G 24 G 25 H 1 H 2 H 3 H 4 H 5 H 6 H 7 H 8 H 9 H 10 H 11 H 12 H 13 H 14

H 15 H 16 H 17 H 18 H 19 H 20 H 21 H 22 H 23 H 24 H 25 I 1 I 2 I 3 I 4 I 5 I 6 I 7 I 8 I 9 I 10

I 11 I 12 I 13 I 14 I 15 I 16 I 17 I 18 I 19 I 20 I 21 I 22 I 23 I 24 I 25 J 1 J 2 J 3 J 4 J 5 J 6

J 7 J 8 J 9 J 10 J 11 J 12 J 13 J 14 J 15 J 16 J 17 J 18 J 19 J 20 J 21 J 22 J 23 J 24 J 25 K 1 K 2

K 3 K 4 K 5 K 6 K 7 K 8 K 9 K 10 K 11 K 12 K 13 K 14 K 15 K 16 K 17 K 18 K 19 K 20 K 21 K 22 K 23

K 24 K 25 L 1 L 2 L 3 L 4 L 5 L 6 L 7 L 8 L 9 L 10 L 11 L 12 L 13 L 14 L 15 L 16 L 17 L 18 L 19

L 20 L 21 L 22 L 23 L 24 L 25 M 1 M 2 M 3 M 4 M 5 M 6 M 7 M 8 M 9 M 10 M 11 M 12 M 13 M 14 M 15

M 16 M 17 M 18 M 19 M 20 M 21 M 22 M 23 M 24 M 25 N 1 N 2 N 3 N 4 N 5 N 6 N 7 N 8 N 9 N 10 N 11

N 12 N 13 N 14 N 15 N 16 N 17 N 18 N 19 N 20 N 21 N 22 N 23 N 24 N 25 O 1 O 2 O 3 O 4 O 5 O 6 O 7

O 8 O 9 O 10 O 11 O 12 O 13 O 14 O 15 O 16 O 17 O 18 O 19 O 20 O 21 O 22 O 23 O 24 O 25 P 1 P 2 P 3

P 4 P 5 P 6 P 7 P 8 P 9 P 10 P 11 P 12 P 13 P 14 P 15 P 16 P 17 P 18 P 19 P 20 P 21 P 22 P 23 P 24

P 25 Q 1 Q 2 Q 3 Q 4 Q 5 Q 6 Q 7 Q 8 Q 9 Q 10 Q 11 Q 12 Q 13 Q 14 Q 15 Q 16 Q 17 Q 18 Q 19 Q 20

Q 21 Q 22 Q 23 Q 24 Q 25 R 1 R 2 R 3 R 4 R 5 R 6 R 7 R 8 R 9 R 10 R 11 R 12 R 13 R 14 R 15 R 16

R 17 R 18 R 19 R 20 R 21 R 22 R 23 R 24 R 25 S 1 S 2 S 3 S 4 S 5 S 6 S 7 S 8 S 9 S 10 S 11 S 12

S 13 S 14 S 15 S 16 S 17 S 18 S 19 S 20 S 21 S 22 S 23 S 24 S 25 T 1 T 2 T 3 T 4 T 5 T 6 T 7 T 8

T 9 T 10 T 11 T 12 T 13 T 14 T 15 T 16 T 17 T 18 T 19 T 20 T 21 T 22 T 23 T 24 T 25 U 1 U 2 U 3 U 4

U 5 U 6 U 7 U 8 U 9 U 10 U 11 U 12 U 13 U 14 U 15 U 16 U 17 U 18 U 19 U 20 U 21 U 22 U 23 U 24 U 25

V 1 V 2 V 3 V 4 V 5 V 6 V 7 V 8 V 9 V 10 V 11 V 12 V 13 V 14 V 15 V 16 V 17 V 18 V 19 V 20 V 21

V 22 V 23 V 24 V 25 W 1 W 2 W 3 W 4 W 5 W 6 W 7 W 8 W 9 W 10 W 11 W 12 W 13 W 14 W 15 W 16 W 17

W 18 W 19 W 20 W 21 W 22 W 23 W 24 W 25 X 1 X 2 X 3 X 4 X 5 X 6 X 7 X 8 X 9 X 10 X 11 X 12 X 13

X 14 X 15 X 16 X 17 X 18 X 19 X 20 X 21 X 22 X 23 X 24 X 25 Y 1 Y 2 Y 3 Y 4 Y 5 Y 6 Y 7 Y 8 Y 9

Y 10 Y 11 Y 12 Y 13 Y 14 Y 15 Y 16 Y 17 Y 18 Y 19 Y 20 Y 21 Y 22 Y 23 Y 24 Y 25

------- test 6 ----

30 26

A 1

A 4 A 5 A 6 A 7 A 8 A 9 A 10 A 11 A 12 A 13 A 14 A 15 A 16 A 17 A 18 A 19 A 20 A 21 A 22 A 23 A 24

A 25 A 26 A 27 A 28 A 29 A 30 B 1 B 2 B 3 B 4 B 5 B 6 B 7 B 8 B 9 B 10 B 11 B 12 B 13 B 14 B 15

B 16 B 17 B 18 B 19 B 20 B 21 B 22 B 23 B 24 B 25 B 26 B 27 B 28 B 29 B 30 C 1 C 2 C 3 C 4 C 5 C 6

C 7 C 8 C 9 C 10 C 11 C 12 C 13 C 14 C 15 C 16 C 17 C 18 C 19 C 20 C 21 C 22 C 23 C 24 C 25 C 26 C 27

C 28 C 29 C 30 D 1 D 2 D 3 D 4 D 5 D 6 D 7 D 8 D 9 D 10 D 11 D 12 D 13 D 14 D 15 D 16 D 17 D 18

D 19 D 20 D 21 D 22 D 23 D 24 D 25 D 26 D 27 D 28 D 29 D 30 E 1 E 2 E 3 E 4 E 5 E 6 E 7 E 8 E 9

E 10 E 11 E 12 E 13 E 14 E 15 E 16 E 17 E 18 E 19 E 20 E 21 E 22 E 23 E 24 E 25 E 26 E 27 E 28 E 29 E 30

F 1 F 2 F 3 F 4 F 5 F 6 F 7 F 8 F 9 F 10 F 11 F 12 F 13 F 14 F 15 F 16 F 17 F 18 F 19 F 20 F 21

F 22 F 23 F 24 F 25 F 26 F 27 F 28 F 29 F 30 G 1 G 2 G 3 G 4 G 5 G 6 G 7 G 8 G 9 G 10 G 11 G 12

G 13 G 14 G 15 G 16 G 17 G 18 G 19 G 20 G 21 G 22 G 23 G 24 G 25 G 26 G 27 G 28 G 29 G 30 H 1 H 2 H 3

H 4 H 5 H 6 H 7 H 8 H 9 H 10 H 11 H 12 H 13 H 14 H 15 H 16 H 17 H 18 H 19 H 20 H 21 H 22 H 23 H 24

H 25 H 26 H 27 H 28 H 29 H 30 I 1 I 2 I 3 I 4 I 5 I 6 I 7 I 8 I 9 I 10 I 11 I 12 I 13 I 14 I 15

I 16 I 17 I 18 I 19 I 20 I 21 I 22 I 23 I 24 I 25 I 26 I 27 I 28 I 29 I 30 J 1 J 2 J 3 J 4 J 5 J 6

J 7 J 8 J 9 J 10 J 11 J 12 J 13 J 14 J 15 J 16 J 17 J 18 J 19 J 20 J 21 J 22 J 23 J 24 J 25 J 26 J 27

J 28 J 29 J 30 K 1 K 2 K 3 K 4 K 5 K 6 K 7 K 8 K 9 K 10 K 11 K 12 K 13 K 14 K 15 K 16 K 17 K 18

K 19 K 20 K 21 K 22 K 23 K 24 K 25 K 26 K 27 K 28 K 29 K 30 L 1 L 2 L 3 L 4 L 5 L 6 L 7 L 8 L 9

L 10 L 11 L 12 L 13 L 14 L 15 L 16 L 17 L 18 L 19 L 20 L 21 L 22 L 23 L 24 L 25 L 26 L 27 L 28 L 29 L 30

M 1 M 2 M 3 M 4 M 5 M 6 M 7 M 8 M 9 M 10 M 11 M 12 M 13 M 14 M 15 M 16 M 17 M 18 M 19 M 20 M 21

M 22 M 23 M 24 M 25 M 26 M 27 M 28 M 29 M 30 N 1 N 2 N 3 N 4 N 5 N 6 N 7 N 8 N 9 N 10 N 11 N 12

N 13 N 14 N 15 N 16 N 17 N 18 N 19 N 20 N 21 N 22 N 23 N 24 N 25 N 26 N 27 N 28 N 29 N 30 O 1 O 2 O 3

O 4 O 5 O 6 O 7 O 8 O 9 O 10 O 11 O 12 O 13 O 14 O 15 O 16 O 17 O 18 O 19 O 20 O 21 O 22 O 23 O 24

O 25 O 26 O 27 O 28 O 29 O 30 P 1 P 2 P 3 P 4 P 5 P 6 P 7 P 8 P 9 P 10 P 11 P 12 P 13 P 14 P 15

P 16 P 17 P 18 P 19 P 20 P 21 P 22 P 23 P 24 P 25 P 26 P 27 P 28 P 29 P 30 Q 1 Q 2 Q 3 Q 4 Q 5 Q 6

Q 7 Q 8 Q 9 Q 10 Q 11 Q 12 Q 13 Q 14 Q 15 Q 16 Q 17 Q 18 Q 19 Q 20 Q 21 Q 22 Q 23 Q 24 Q 25 Q 26 Q 27

Q 28 Q 29 Q 30 R 1 R 2 R 3 R 4 R 5 R 6 R 7 R 8 R 9 R 10 R 11 R 12 R 13 R 14 R 15 R 16 R 17 R 18

R 19 R 20 R 21 R 22 R 23 R 24 R 25 R 26 R 27 R 28 R 29 R 30 S 1 S 2 S 3 S 4 S 5 S 6 S 7 S 8 S 9

S 10 S 11 S 12 S 13 S 14 S 15 S 16 S 17 S 18 S 19 S 20 S 21 S 22 S 23 S 24 S 25 S 26 S 27 S 28 S 29 S 30

T 1 T 2 T 3 T 4 T 5 T 6 T 7 T 8 T 9 T 10 T 11 T 12 T 13 T 14 T 15 T 16 T 17 T 18 T 19 T 20 T 21

T 22 T 23 T 24 T 25 T 26 T 27 T 28 T 29 T 30 U 1 U 2 U 3 U 4 U 5 U 6 U 7 U 8 U 9 U 10 U 11 U 12

U 13 U 14 U 15 U 16 U 17 U 18 U 19 U 20 U 21 U 22 U 23 U 24 U 25 U 26 U 27 U 28 U 29 U 30 V 1 V 2 V 3

V 4 V 5 V 6 V 7 V 8 V 9 V 10 V 11 V 12 V 13 V 14 V 15 V 16 V 17 V 18 V 19 V 20 V 21 V 22 V 23 V 24

V 25 V 26 V 27 V 28 V 29 V 30 W 1 W 2 W 3 W 4 W 5 W 6 W 7 W 8 W 9 W 10 W 11 W 12 W 13 W 14 W 15

W 16 W 17 W 18 W 19 W 20 W 21 W 22 W 23 W 24 W 25 W 26 W 27 W 28 W 29 W 30 X 1 X 2 X 3 X 4 X 5 X 6

X 7 X 8 X 9 X 10 X 11 X 12 X 13 X 14 X 15 X 16 X 17 X 18 X 19 X 20 X 21 X 22 X 23 X 24 X 25 X 26 X 27

X 28 X 29 X 30 Y 1 Y 2 Y 3 Y 4 Y 5 Y 6 Y 7 Y 8 Y 9 Y 10 Y 11 Y 12 Y 13 Y 14 Y 15 Y 16 Y 17 Y 18

Y 19 Y 20 Y 21 Y 22 Y 23 Y 24 Y 25 Y 26 Y 27 Y 28 Y 29 Y 30 Z 1 Z 2 Z 3 Z 4 Z 5 Z 6 Z 7 Z 8 Z 9

Z 10 Z 11 Z 12 Z 13 Z 14 Z 15 Z 16 Z 17 Z 18 Z 19 Z 20 Z 21 Z 22 Z 23 Z 24 Z 25 Z 26 Z 27 Z 28 Z 29 Z 30

------- test 7 ----

2 26

A 1

Y 1

------- test 8 ----

10 10

E 10

A 10 I 4 I 6 C 2 I 10 G 3 J 6 E 10 B 1 J 9 A 5 F 10 J 8 E 1 A 1 E 7 J 7 D 5 G 5 H 1 J 4

C 6 E 3 G 6 A 3 J 1 H 2 D 10 G 1 C 7 A 6 G 7 F 6 I 1 I 2 A 2 J 2 G 4 D 1 G 2 A 7 D 7

G 9 I 5 F 1 D 9 J 3 I 3 F 2 B 6 E 2 C 3 F 8 D 2 C 9 B 7 B 4 B 8 H 3 B 2 D 4 B 9 H 7

I 7 E 9 E 4 F 3 F 5 I 9 H 6 J 10 H 4 C 1 C 8 A 9 F 4 H 10 A 8 C 5 B 10 E 5 C 4 C 10 B 5

F 9 E 8 A 4 I 8 J 5 H 8

------- test 9 ----

20 20

I 1

T 5 B 5 J 11 E 3 A 17 H 17 E 20 D 2 J 13 P 16 R 15 G 16 D 9 B 20 C 8 C 7 J 16 R 2 H 16 R 5 P 15

M 2 N 8 I 2 I 18 G 17 P 18 R 4 H 8 A 16 F 14 O 12 D 8 P 1 Q 19 O 14 G 19 B 4 M 7 E 13 T 3 Q 2

G 5 Q 1 K 5 I 14 S 18 P 10 J 20 A 19 R 12 C 11 N 1 Q 20 J 17 I 8 H 1 A 18 J 19 A 9 A 3 H 12 K 11

N 10 T 19 A 10 T 20 S 11 O 11 B 15 J 9 L 11 N 6 E 17 F 9 M 1 Q 8 C 3 C 6 F 19 O 15 F 17 M 10 A 15

P 19 P 13 T 15 K 8 N 2 S 10 J 1 N 13 G 14 L 16 E 6 I 20 E 11 P 20 I 12 O 19 J 6 E 18 T 16 M 15 J 10

D 17 T 12 F 7 L 8 L 14 R 16 H 20 Q 13 R 20 C 1 F 18 M 18 I 13 F 3 K 2 G 6 H 2 S 15 Q 11 T 17 S 5

E 8 T 18 L 20 M 6 H 14 M 4 P 17 P 11 P 12 E 1 Q 5 P 4 R 11 G 7 R 17 B 17 B 9 R 9 M 5 H 13 K 14

A 6 B 6 B 19 G 15 S 8 H 6 C 13 J 18 F 12 G 3 E 12 I 6 C 5 D 11 M 12 C 19 C 17 E 7 S 12 Q 16 A 12

B 18 F 6 J 5 C 4 I 11 G 9 J 2 G 13 N 18 J 12 D 15 G 12 P 2 M 8 I 15 G 1 J 8 Q 7 B 7 Q 17 O 1

A 8 H 19 E 2 T 14 N 5 I 5 N 3 G 18 O 6 F 11 Q 10 S 19 E 4 O 20 H 5 K 4 E 16 Q 18 N 11 M 17 A 20

A 7 E 10 T 11 P 8 D 19 D 16 G 4 J 7 S 7 Q 3 K 13 F 15 D 6 E 19 D 20 S 4 K 17 R 7 T 10 M 13 D 10

K 19 K 3 A 1 M 9 G 2 I 17 G 11 K 15 N 16 L 13 C 10 P 9 T 13 S 16 Q 6 L 5 C 14 E 15 B 10 F 10 D 1

L 10 M 3 E 14 H 3 J 4 M 16 A 13 I 16 B 13 D 5 J 15 M 11 F 8 S 6 K 18 S 2 O 8 A 4 O 3 O 16 C 18

H 9 D 18 P 5 L 17 L 15 S 17 B 2 C 2 L 7 R 10 D 3 O 13 A 14 R 18 Q 9 C 20 C 12 D 12 M 19 T 6 L 4

F 2 Q 12 T 7 F 16 R 1 I 3 B 1 G 10 H 15 K 10 O 10 L 1 N 12 S 13 A 2 F 4 N 9 M 14 N 4 L 19 C 15

N 7 L 6 O 9 T 4 F 13 S 14 T 8 Q 15 S 9 O 7 M 20 P 7 Q 4 R 14 R 3 O 17 S 20 K 16 P 6 F 5 H 18

R 19 J 14 D 14 C 16 K 9 D 13 A 5 K 7 K 6 T 1 G 8 Q 14 B 12 O 5 H 11 L 12 I 19 F 1 H 4 B 8 L 9

E 9 B 16 I 7 I 9 O 4 N 20 A 11 I 10 D 7 O 18 R 8 B 11 H 10 P 14 B 14 N 14 I 4 P 3 T 9 K 12 S 1

C 9 O 2 L 18 I 1 L 3 N 19 G 20 T 2 H 7 S 3 K 1 J 3 K 20 L 2 N 15 B 3 E 5 D 4 R 13 N 17 F 20

R 6

------- test 10 ----

30 26

K 2

F 7 H 5 M 14 W 5 B 24 C 6 C 25 Q 6 D 9 T 28 P 28 W 22 F 2 G 5 F 12 N 18 R 20 E 26 C 18 O 1 B 10

U 23 G 16 J 1 C 4 M 16 Z 4 P 18 V 1 B 13 U 20 H 1 W 19 T 1 L 14 S 2 N 21 X 6 N 19 H 13 B 2 A 13

B 28 D 13 Y 17 Q 12 B 3 M 9 K 14 R 19 R 23 M 28 D 8 B 18 M 5 U 29 Q 25 P 8 O 22 O 25 V 6 P 2 D 2

N 30 K 29 L 27 F 10 U 19 B 19 I 18 K 5 P 21 B 26 C 12 T 20 T 17 M 22 H 8 S 26 C 19 O 19 Y 4 Q 9 G 28

S 3 Q 11 V 4 F 30 A 14 N 3 B 11 R 29 I 27 P 16 T 14 D 3 X 3 S 18 X 22 J 6 P 26 B 12 T 3 K 20 U 4

E 4 D 17 F 18 V 18 W 8 D 26 C 11 S 23 X 14 Z 18 K 19 S 8 D 15 Z 10 C 5 P 17 P 6 Z 1 T 27 T 2 F 15

H 14 N 17 Q 27 Z 19 Q 4 M 7 J 9 L 5 H 25 O 7 E 3 Z 28 A 10 X 11 Z 30 L 2 G 23 G 8 S 19 A 4 A 21

W 12 B 30 I 2 L 29 P 30 B 15 V 10 K 24 N 11 O 17 E 22 A 30 T 9 L 6 X 8 J 24 D 10 J 14 W 9 N 14 Y 26

U 13 W 27 F 22 V 16 I 24 B 14 U 14 J 21 S 14 G 2 Y 28 R 28 E 6 I 19 C 14 O 23 F 8 D 28 K 7 H 21 O 12

E 16 U 3 I 13 N 4 I 14 V 15 Z 13 E 20 Z 12 B 9 I 22 L 15 X 18 U 30 C 7 S 28 W 30 L 4 Y 10 O 9 N 16

Y 20 E 21 Y 22 H 10 O 14 M 27 O 11 B 25 G 13 K 16 S 24 O 4 V 8 R 22 C 13 D 5 K 17 G 26 A 8 P 20 H 15

J 7 O 18 T 8 J 20 Z 17 W 17 I 3 R 9 N 24 B 5 L 7 J 18 Y 24 J 11 W 24 P 7 M 19 I 20 Q 18 W 10 G 25

Q 24 A 17 L 12 R 1 Q 19 A 25 F 24 E 1 K 28 N 26 R 26 B 4 R 27 Z 20 F 28 Z 5 I 21 F 23 L 16 R 13 A 19

L 26 Z 21 F 6 E 25 V 29 S 7 R 7 O 30 Z 16 U 17 P 11 G 3 V 5 U 25 T 5 H 7 A 6 J 13 B 1 I 17 O 16

Q 23 R 5 V 17 F 25 E 27 M 29 W 26 L 10 F 21 G 12 C 28 A 20 O 6 O 29 Q 16 Z 26 X 12 E 18 M 30 M 2 E 2

F 9 G 21 C 10 J 26 Y 9 Q 30 P 4 Q 2 P 27 R 6 N 12 X 26 L 25 S 10 U 12 Q 17 N 25 L 18 I 26 W 16 D 1

I 23 E 13 U 2 W 4 C 1 S 20 V 3 Q 29 L 3 X 29 J 4 Y 13 K 22 Y 2 W 14 V 24 S 6 X 15 C 16 O 28 T 21

H 12 A 23 T 16 R 8 U 16 V 19 Y 11 T 10 M 18 E 7 M 3 S 21 K 2 I 11 Q 3 U 22 R 15 M 4 R 30 H 22 Y 5

F 26 E 23 U 6 B 16 B 21 K 27 R 3 W 21 O 3 Y 18 Y 1 O 10 H 17 Q 1 D 21 A 26 V 13 X 13 G 7 I 1 C 21

H 30 Z 7 I 5 N 6 C 8 M 1 N 10 W 18 E 11 U 5 T 24 O 8 G 22 L 17 J 28 J 3 Y 16 K 21 Q 13 Y 3 F 27

T 11 U 24 I 8 E 30 K 8 F 19 S 16 X 16 R 10 J 30 G 24 Y 14 N 29 R 16 P 23 S 30 O 2 P 5 D 7 Q 26 L 28

Z 25 F 14 G 20 A 29 L 13 U 9 D 30 P 24 J 22 Q 20 O 26 W 25 M 26 M 24 G 1 T 13 N 28 Y 12 E 28 D 23 H 24

V 7 B 29 W 15 Y 25 H 26 R 12 R 25 C 15 D 12 U 28 Q 14 G 6 U 15 X 4 E 15 Z 9 X 10 H 20 P 22 D 27 L 30

U 8 Y 21 W 2 E 12 I 30 J 8 L 24 L 11 C 20 Y 23 R 4 N 27 W 13 D 18 A 24 Z 2 O 27 N 9 E 10 W 7 P 1

T 18 A 2 A 3 D 24 C 23 V 21 V 22 C 22 X 30 U 1 N 7 D 22 V 14 O 24 K 10 B 20 Y 15 M 11 H 23 M 6 O 20

T 30 E 14 Q 15 O 15 T 15 M 17 A 12 H 11 H 9 V 12 C 2 H 27 L 8 Z 14 T 25 E 19 A 28 H 19 S 12 A 16 I 4

O 21 F 11 K 23 Q 22 K 25 P 3 L 1 L 19 E 8 K 1 M 21 M 13 V 28 X 1 B 27 V 9 X 2 Q 10 G 9 J 5 V 30

P 9 T 12 T 19 R 17 G 19 A 18 Y 8 Z 8 K 15 S 25 C 17 X 21 P 19 K 6 E 24 T 6 R 24 G 15 B 17 P 15 P 12

J 19 X 23 I 10 Q 5 K 4 A 22 W 3 M 12 I 6 U 18 K 30 C 26 S 4 T 22 B 6 J 27 A 15 E 9 W 23 H 2 A 1

V 2 M 8 T 7 H 4 Y 29 K 18 H 28 C 29 I 12 J 25 X 7 E 17 L 21 B 23 N 5 R 14 J 29 G 4 X 9 X 20 H 16

S 29 O 13 X 5 M 20 F 16 X 17 M 23 Z 11 R 18 W 29 A 7 I 7 V 25 A 11 F 17 N 13 G 11 E 29 F 3 G 27 D 25

I 25 M 15 R 21 U 26 J 23 U 10 Q 28 I 28 V 23 R 11 Y 6 I 9 G 17 A 5 E 5 D 11 H 6 Z 6 S 15 H 3 J 15

Z 15 P 13 C 9 K 11 O 5 S 5 H 29 G 18 M 10 S 27 B 22 I 16 U 21 V 11 A 27 K 13 W 11 F 1 T 23 K 26 U 7

B 8 K 3 Y 27 K 12 Y 19 Z 27 D 4 Y 30 I 29 L 23 W 28 Z 22 P 29 N 23 T 26 N 1 X 25 D 6 H 18 W 1 J 16

Q 7 B 7 G 30 F 13 D 16 N 2 A 9 Z 3 Q 21 K 9 N 22 S 17 X 19 T 29 F 4 Z 23 J 10 D 14 Z 24 N 8 V 27

D 20 P 14 X 27 N 15 C 24 J 12 F 29 G 29 C 30 D 19 P 25 F 20 W 20 P 10 T 4 L 20 W 6 D 29 J 17 F 5 V 20

C 27 C 3 L 22 X 24 R 2 U 27 G 10 J 2 G 14 S 22 S 1 M 25 L 9 X 28 S 9 Y 7 S 11 Q 8 V 26 I 15 S 13

U 11 N 20 Z 29

------- test 11 ----

30 4

B 15

A 1 B 1 C 1 D 1 A 30 B 30 C 30 D 30 A 15 C 15

------- test 12 ----

8 8

A 2

C 1

------- test 13 ----

8 8

D 5

A 3 A 8 H 1 H 8

------- test 14 ----

8 8

A 8

B 6 B 7 C 7

------- test 15 ----

8 8

A 1

A 2 A 3 H 2 H 5 H 6 H 7 H 8 C 1 C 2 C 5 C 6 C 7 E 2 E 3 E 4 E 5 E 6 E 7 E 8

------- test 16 ----

8 8

A 1

A 2 A 3 A 4 A 5 A 6 A 7 A 8 H 1 H 2 H 3 H 4 H 5 H 6 H 7 H 8 G 1 G 2 G 3 C 1 C 2 C 3

C 4 C 5 C 6 C 7 C 8 E 1 E 2 E 3 E 4 E 5 E 6 E 7 E 8 F 1 F 2 F 3 F 4 F 5 F 6 F 7 F 8

D 1 D 2 D 3 D 4 D 5 D 6 D 7 D 8 B 1 B 2 B 3 B 4 B 5 B 6 B 7 B 8

------- test 17 ----

3 3

B 2

A 1 A 2 A 3 B 1 B 3 C 1 C 2 C 3

------- test 18 ----

2 2

A 1

B 2

------- test 19 ----

8 8

G 4

E 4 D 6

------- test 20 ----

8 8

D 5

B 1

F 1

B 3

Keep up the good work!

Thanks for your submission!

[Problem Statement for task camelot](http://ace.delos.com/usacoprob2?S=camelot&a=aFfaH0aWT0T) | [USACO Gateway](http://ace.delos.com/usacogate?a=aFfaH0aWT0T) | [Comment or Question](mailto:usaco@ace.delos.com)

http://ace.delos.com/clear_dot

http://ace.delos.com/clear_dot

窗体顶端

**Submit a file:** 

窗体底端

/\*

ID:nealgav1

LANG:C++

PROG:camelot

\*/

#include<fstream>

#include<cstring>

#include<queue>

using namespace std;

ifstream cin("camelot.in");

ofstream cout("camelot.out");

const int mm=31;

const int oo=1e9;

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

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

int R,C;

class node

{

public:int r,c;

void data(int a,int b){r=a;c=b;}

}knigh[mm\*mm],king;

int dis[mm][mm],kkdis[mm][mm],kdis[mm][mm],cost[mm][mm],kcost[mm][mm];

bool vis[mm][mm];

int pos;

queue<node>Q;

void bfs(node night)

{ node d;int x,y;

memset(vis,0,sizeof(vis));

for(int i=0;i<R;i++)

for(int j=0;j<C;j++)

dis[i][j]=oo;

Q.push(night);dis[night.r][night.c]=0;

while(!Q.empty())

{ d=Q.front();Q.pop();

vis[d.r][d.c]=1;

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

{ x=d.r+dx[i];

y=d.c+dy[i];

if(x>=0&&x<R&&y>=0&&y<C&&!vis[x][y])

{

dis[x][y]=dis[d.r][d.c]+1;

node s;s.data(x,y);

Q.push(s);

}

}

}

for(int i=0;i<R;i++)

for(int j=0;j<C;j++)

cost[i][j]+=dis[i][j];//knight

for(int i=0;i<R;i++)

for(int j=0;j<C;j++)

kkdis[i][j]=kdis[i][j];

for(int i=0;i<R;i++)

for(int j=0;j<C;j++)

{ memset(vis,0,sizeof(vis));

node d;d.data(i,j);

Q.push(d);

while(!Q.empty())

{

d=Q.front();Q.pop();

vis[d.r][d.c]=1;

for(int k=0;k<8;k++)

{

x=d.r+dx[k];

y=d.c+dy[k];

if(x>=0&&x<R&&y>=0&&y<C&&!vis[x][y])

{

if(dis[x][y]+kkdis[x][y]>dis[d.r][d.c]+kkdis[d.r][d.c]+1)

{

kkdis[x][y]=dis[d.r][d.c]+kkdis[d.r][d.c]+1-dis[x][y];

}

node s;s.data(x,y);

Q.push(s);

}

}

}

}

for(int i=0;i<R;i++)

for(int j=0;j<C;j++)

if(kcost[i][j]>kkdis[i][j])

kcost[i][j]=kkdis[i][j];

}

int \_abs(int x)

{

return x>0?x:-x;

}

int main()

{ char a,b;

while(cin>>C>>R)

{

cin>>a>>b;

memset(cost,0,sizeof(cost));

king.data(a-'A',b-'1');

for(int i=0;i<R;i++)

for(int j=0;j<C;j++)

kcost[i][j]=kdis[i][j]=\_abs(i-king.r)>\_abs(j-king.c)?\_abs(i-king.r):\_abs(j-king.c);

pos=0;

while(cin>>a>>b)

{

knigh[pos++].data(a-'A',b-'1');

bfs(knigh[pos-1]);

}

int ans=oo;

for(int i=0;i<R;i++)

for(int j=0;j<C;j++)

if(kcost[i][j]+cost[i][j]<ans)ans=kcost[i][j]+cost[i][j];

cout<<ans<<"\n";

}

}