# Exit Point in a Matrix

1. You are given a number n, representing the number of rows.  
2. You are given a number m, representing the number of columns.  
3. You are given n\*m numbers (1's and 0's), representing elements of 2d array a.  
4. Consider this array a maze and a player enters from top-left corner in east direction.  
5. The player moves in the same direction as long as he meets '0'. On seeing a 1, he takes a 90 deg right turn.  
6. You are required to print the indices in (row, col) format of the point from where you exit the matrix.

Input Format

A number n  
A number m  
e11  
e12..  
e21  
e22..  
.. n \* m number of elements

Output Format

row  
col (of the point of exit)

void fill2DMatrix(*vector*<*vector*<int>>& matrix){

int ele, rows = matrix.*size*(), cols = matrix[0].*size*();

for (auto i = 0; i < rows; i++){

for (auto j = 0; j < cols; j++){

*cin* >> ele;

matrix[i][j] = ele;

}

}

}

#define NUMBER\_OF\_DIRECTIONS 4

int main(){

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

*vector*<*vector*<int>> matrix(n, *vector*<int>(m));

fill2DMatrix(matrix);

auto dir = 0;

auto i = 0, j = 0;

while (true) {

dir = (dir + matrix[i][j]) % NUMBER\_OF\_DIRECTIONS;

if (dir == 0) // Moving East

j++;

else if (dir == 1) // Moving South

i++;

else if (dir == 2) // Moving West

j--;

else // Moving North

i--;

if (i < 0) {

i++;

break;

}

else if (j < 0) {

j++;

break;

}

else if (i == matrix.*size*()) {

i--;

break;

}

else if (j == matrix[0].*size*()) {

j--;

break;

}

}

*cout* << i << " " << j << *endl*;

return 0;

}