Test Description

You are given a board of N rows and M columns. Each field of the board contains a single digit (0-9).

You want to find a path consisting of four neighboring fields. Two fields are neighboring if they share a common side. Also, the fields in your path should be distinct (you can't visit the same field twice).

The four digits of your path, in order in which you visit them, create an integer. What is the biggest integer that you can achieve in this way?

Write a class with this entry function, submit the class file to us after completion.

```
class Solution
{
      public int solution(int[][] Board)
      {
            //Your code goes here
      }
}
```

That, given the board represented as a matrix of integers consisting of N rows and M columns, returns the biggest integer that you can achieve when concatenating the values in a path of length four.

Examples:

1. Given the following board (N=3, M=5)

9	1	1	0	7
1	0	2	1	0
1	9	1	1	0

   The function should return 9121. You can choose either of the following paths (the first field is denoted by red):

9	1	1	0	7
1	0	2	1	0
1	9	1	1	0

   Or

9	1	1	0	7
1	0	2	1	0
1	9	1	1	0

2. Given the following board (N=3, M=3):

1	1	1
1	3	4
1	4	3

The function should return 4343. The best path is:

1	1	1
1	3	4
1	4	3

3. Given the following board (N=1, M=5):

0	1	5	0	0

the function should return 1500. The best path is:

0	1	5	0	0

Write an efficient algorithm for the following assumptions:

- N and M are integers within the range [1…100];
- each element of matrix Board is an integer within the range [0…9];
- there exists a path of length 4 which doesn't start with 0.