Shown below is the complete output of the provided input:

INPUT: (Team 39)

```
--------------------------------------
4 4
-39 0 0 0
0 0 0 0
3.9 0 0 0
0 -3.9 0 39
2 1
0 0
3 3
1 2
3 0
-7.8
```

OUTPUT:

```
--------------------------------------
          ITERATION-0

['O', 'O', 'O', 'O']
['O', 'O', 'O', 'O']
['O', 'O', 'O', 'O']
['O', 'O', 'O', 'O']

[-39.0, 0.0, 0.0, 0.0]
[0.0, 0.0, 0.0, 0.0]
[3.9, 0.0, 0.0, 0.0]
[0.0, -3.9, 0.0, 39.0]
--------------------------------------
          ITERATION-1

['O', 'E', 'E', 'E']
['S', 'S', 'O', 'S']
['W', 'E', 'E', 'S']
['S', 'E', 'E', 'O']

[-39.0, -7.8, -7.8, -7.8]
[-4.68, -8.268, 0.0, -7.8]
[-5.148, -9.017, -7.8, 22.62]
[-8.19, -9.092, 22.62, 39.0]
```

Change:

We see a change in policy. Before any of the iterations
we have a random policy, i.e. nothing is fixed and hence
at the end of the first iteration, we see that on the basis
of the neighbouring(next) states, the policies of the states
are formed.

For example, state(0,1) tries to move away from the state(0,0)
and hence it has two possibilities S or E. But as, it has

an uncertainity to go towards (0,0) if S, hence it selects E.

Similarly, on the basis of very next neighbours, the policies
of all the states are decided, and wherever there are more than
one choice of policy, we select on the basis of FCFS(N-E-W-S).

ITERATION-2

['O', 'E', 'E', 'E']
['S', '**E**', 'O', 'S']
['W', 'E', 'E', 'S']
['S', 'E', 'E', 'O']

[-39.0, -15.647, -15.6, -15.6]
[-13.213, -16.881, 0.0, 8.736]
[-14.059, -16.637, 11.778, 26.84]
[-16.08, 7.723, 26.84, 39.0]

Change:

Here, the policy of (1,1) changed from S to E. In the iteration-1,
the values of the immediate neighbours of (1,1), other than N were
0, annd hence it selected S because it had to account only for the
0.1 uncertainty towards (1,0). But, at the end of the iteration, the
values were updated and hence, the policy was changed considering
the far rewards.

Also, in this iteration, if it selects E, then it has a 0.8 certainty
to stay in the same block which has the best utility at the time
instant, and hence it selects E over S, i.e to stay in the same block,
but it can be assumed that this policy too would change with the
coming iterations as it won't get any better because of the negative
value of the step cost.

ITERATION-3

['O', 'E', 'E', 'S']
['S', '**S**', 'O', 'S']
['W', 'E', 'E', 'S']
['**E**', 'E', 'E', 'O']

[-39.0, -23.533, -23.4, -4.711]
[-22.056, -25.004, 0.0, 15.419]
[-22.861, -0.106, 17.534, 27.837]
[-5.516, 14.434, 27.837, 39.0]

Change:

The policy of (1,1) and (3,0) has changed from E to S and S to E
respectively.

Policy of (1,1) - As highlighted in the last iteration, that it accepted to stay in the same state which would not reap better utilities in further iterations. Hence, in this iteration, the policy changed from E to S. It is because, in the previous iteration, the utility of (2,1) wasn't updated and hence we observed that at the end of the previous iteration, the utility of (2,1) imporved over (1,1) which holds the 0.8 certainity if the policy is selected to be S or E respectively. Thus, in this iteration, we end up selecting S over E as the utility of (2,1) is marginally better than (1,1). We also need to take into consideration of the 0.1 uncertainty in the perpendicular directions, but as it doesn't affect much in this case, we don't need to consider it as the cause.

Policy of (3,0) - Same as above. Till the previous iteration, it selected to stay in the same block owing to the more negative value of (3,1). But, in the previous iteration it was updated to be significantly positive because it prefers the policy to move E towards the end terminal state having utility [39]. And hence, in this iteration, the policy of (3,0) is updated to E from S, because the utility was updated to a positive value while the utility remained negative for (3,0).

ITERATION-4

['O', 'E', 'E', 'S']
['S', 'S', 'O', 'S']
['**E**', 'E', 'E', 'S']
['E', 'E', 'E', 'O']

[-39.0, -31.374, -16.249, 2.439]
[-30.794, -13.464, 0.0, 17.554]
[-11.516, 6.324, 19.007, 28.084]
[2.044, 16.546, 28.084, 39.0]

Change:

Policy of (2,0) changed from W to E. As mentioned a couple of times before as well that the policy to remain in the same state would be changed to a feasible move towards the state (3,3) in further iterations. Here, initially, the policy was W, because it was the better than (2,1) by 2.5 units. But, the utility of (2,0) is calculated on the basis of the old grid utility and hence when the value wasa updated in the next iteration, it found (2,1), i.e E much better than W, and that too by around 20 units. Hence, we can see that the utility of (2,1) improved significantly with every iteration and so did the utility of (2,0) as it started selecting E as its policy which is far better than W.

ITERATION-5

['O', '**S**', 'E', 'S']
['S', 'S', 'O', 'S']
['E', 'E', 'E', 'S']
['E', 'E', 'E', 'O']

[-39.0, -24.096, -9.098, 5.577]

```
[-21.438, -6.231, 0.0, 18.178]
[-4.68, 8.437, 19.377, 28.146]
[5.173, 17.166, 28.146, 39.0]
```

Change:

The policy for (0,1) changed from E to S. With the passing iterations, we select policies looking at farther benefits, and hence more accurate policies looking at long term benefits. Hence, we find that S gives marginally better profits than E by 1.4 units. Hence, we updated the policy from E to S. But, to keep in importance is the fact that the values of (0,1) are changing like a upwards parabola. First, it decreased and now it has started to increase. Till, the previous iteration, the utility of (0,1) has dropped, but now due to updating the utilities of (0,2) and (1,1), we get a better view of their neighbours and so on. Also, to notice that utility of (0,2)[-23.4] was greater than that of (1,1)[-25.0] in last-to-last iteration, which is changed. (1,1)[-13.56] has better utility than (2,0) [-16.25] in the previous iterations which accounts for the policy of (0,1) in this iteration. It is because the utility of (2,1) improved by higher margins which started affecting the utility of (1,1) and eventually changed the policy of (0,1). In this way, with passing iterations, we have a wider view of all the world states.

ITERATION-6

```
['O', 'S', 'E', 'S']
['S', 'S', 'O', 'S']
['E', 'E', 'E', 'S']
['E', 'E', 'E', 'O']

[-39.0, -17.595, -5.158, 6.785]
[-14.311, -3.105, 0.0, 18.353]
[-1.964, 9.107, 19.469, 28.162]
[6.254, 17.344, 28.162, 39.0]
```

ITERATION-7

```
['O', 'E', 'E', 'S']
['S', 'S', 'O', 'S']
['E', 'E', 'E', 'S']
['E', 'E', 'E', 'O']

[-39.0, -13.996, -3.404, 7.22]
[-11.113, -1.936, 0.0, 18.4]
[-1.0, 9.316, 19.492, 28.165]
[6.601, 17.395, 28.165, 39.0]
```

Change:

The policy of (0,1) has been updated from S to E. As
highlighted above, we noticed the values of (0,1)
increasing after reaching its minimum value. Hence,
inspite of (1,1) having better utility than (0,2),
we select policy E, which says that the uncertainity
of 0.1 plays and important role in selecting the policy
which wasn't expected till this change.

We see that, in policy E, it has an uncertainty towards
N(0,1)[-17.595] and S(1,1)[-3.105] whose values are significantly
lower than that in S, where there is an uncertainity towards
E(0,2)[-5.19] and W(0,0)[-39.0]. Hence, we notice a change in
policy inspite of (1,1) having a better utility than (0,2)
because in MDPs, uncertainity can change policies which is
given importance here.

ITERATION-8

```
['O', 'E', 'E', 'S']
['S', 'S', 'O', 'S']
['E', 'E', 'E', 'S']
['E', 'E', 'E', 'O']

[-39.0, -12.116, -2.705, 7.371]
[-9.905, -1.531, 0.0, 18.412]
[-0.677, 9.38, 19.498, 28.166]
[6.709, 17.41, 28.166, 39.0]
```

ITERATION-9

```
['O', 'E', 'E', 'S']
['S', 'S', 'O', 'S']
['E', 'E', 'E', 'S']
['E', 'E', 'E', 'O']

[-39.0, -11.329, -2.444, 7.423]
[-9.486, -1.397, 0.0, 18.416]
[-0.574, 9.4, 19.5, 28.167]
[6.741, 17.414, 28.167, 39.0]
```

This is the final utility table along with the policies as
mentioned above.

END
