PART 2:TASK 1(TEAM-52[Kartik(2019101060),Ayush(2019111026)])

The output in the trace file is some sort of logical. Like for the different scenarios:

WEST => if i have a single arrow and my Monster’s health is 25 or 75 then the policy says to attack the monster by shooting the arrow, doesn’t matter what the state of MM is. It makes sense for the 25 health part completely. For health = 75, we can relate that the code is trying to reduce it to 50 and then use the blade to attack maybe.

EAST => For each of the possible state of east the policy suggests to HIT the MM using the blade or use the arrows(if we have) to attack the monster. This makes sense because as we are now on a position where we have highest probability to attack so we should attack the monster. One may think that he should go to craft arrows but that is kinda expensive operation to do as we will require to go back to center then gather material then to craft (-40 straight if we go like E -> C -> S -> C -> N, step cost = -10)

CENTER => For center what the policy suggests is to shoot if we have a single arrow and the state of monter does not matter here. But if we have more than one arrow then the policy suggests to shoot if the monster is in DORMANT state else the policy says to save the arrows we have as on attack of MM we will lose all our arrows and they will be of no use to us. So in the center state it either says us to go left or go down to gather material. One can also go to NORTH directly if we already have materials to our full capacity.

NORTH => Stay in north if the monster is in ready state and keep crafting. If we cant craft and the monster is in ready state then stay in here. Else go the south and gather material then again come on north and craft. In midway it can happen that any of the action gets failed and you can end up at east. In this case repeat hitting the monster till it dies.

SOUTH => The policy is similar to north but instead of crafting we have gather in south.

Rate of Convergence: 2.37 [The average of the maximum change in utility in each iteration until convergence of VI]

Iterations: 117 iterations (0 based indexing)

TASK 2.1

WEST: For west the policy for this task says that if numofarrows*25 >= healthMM then choose fire option (same as before), else choose STAY if MM is in ready state(Different, as before in some cases firing was preferred over staying even when MM was ready). Otherwise move the center (as it does increase the chance of hitting MM, moreover the ways to craft and gather are also open now.

NORTH: Now if MM is ready then always choose either stay or craft (avoided if we already have 3 arrows). In case we have 3 arrows and MM is dormant then go to center and use some arrows. Now if MM is dormant and we have materials and no arrows then crafting is chosen as an option. In case we have arrows then go to center spend some. Now if we lack on materials then go to south via center and gather material. In case the MM is ready then choose stay. {This is pretty much the same as before}

EAST: For east what IJ prefers is to keep shooting(in case he has arrows), or to keep hitting the monster with the blade. But in case the monster was in ready state with a health of greator than equal to 75 and IJ had 3 arrows then IJ prefers to go LEFT (Hopefully he aims to go to west so that he can save his arrows and use them later on to attack MM) {This is the difference as now on left move he will directly reach west and he will be able to save his arrows so this change is observed}

SOUTH: South policy is almost same to the previous one. No significant changes are observed when compared with task 2

CENTER: Again similar to before no significant changes observed. The significant change was only observed in the EAST state policy and that is kind of expected as we have changed a move for this state only. Rest all things remain pretty much the same.

Rate of Convergence: 2.248 [The average of the maximum change in utility in each iteration until convergence of VI]

Iterations: 119(0 based indexing)

The rate is supposed to be lesser than before as now the STAY action will be chosen more number of times. The progress is now kinda slow.

TASK 2.2

WEST: A significant change is observed in this task. Now except the states in which MM has 25 health and IJ has an arrow the preferred action is to stay always. But in the case mentioned above it is to shoot and go to end state.[A significant change is oberved , it also makes the set of actions for the start state to be infinitely long as it always prefers to stay in the west state]

NORTH: The policy for North is same and it does not experience any significant change.

EAST: one of the major changes in the policy for the east is that now HIT action can’t be found in the policy given by VI. Instead of that the VI says to choose LEFT action and go to center, probably due to the fact that West state offers zero rather than a negative utility.

SOUTH: In south, we can now barely find the action of gather in the policy. Instead it says to opt for either STAY or go UP. [This state has experienced a major change]. An obvious reason is because now stay is preferred as even after gathering one has to go to north to bring the gathered material to some use. So that implies going center and then to north then crafting and the initial gathering that is in total 4 steps that is -40 cost in total(step cost = -10). But if we just stay the cost is zero.

CENTER: Instead of right now the policy in this step says to go the left and reach WEST. The reason is similar to as given for South position.

Rate of convergence: 1.90

Iterations: 56
The number of iterations has reduced significantly now as for the West state the utility remains always zero rather than negative so each state most of the times find a way to reach West and then they achieve kind of constancy as WEST dont change. This is also evident in the actions suggested for the given two start states.

TASK 2.3

WEST: Now in WEST, what we observe is that we either get shoot or stay. The difference is lies in the part that now VI does not suggest to choose right action as it does not provide any good reward in near future. Instead of that it chooses to stay.

EAST and NORTH: No significant changes observed

SOUTH: The stay action is completely avoided in this position. It either goes UP or gathers. Up in the cases in which IJ can attack which will result in some kind of reward as the MM health will decrease in case IJ attacks.

CENTER: The diffrence is that in the initial policy we used to opt for right and then repeatedly use SHOOT and HIT actions in order to reach end state. Since these SHOOT and HIT can go for many steps like they may fail or MM can attack so which results in the fact that these states may not provide good results in near future although in the long run they are guranteed to provide so. Since we now had gamma = 0.25 that is we are weighing the neat rewards much more than the later rewards. So we either choose shoot and decrease MM’s health or to go WEST and save his arrows.

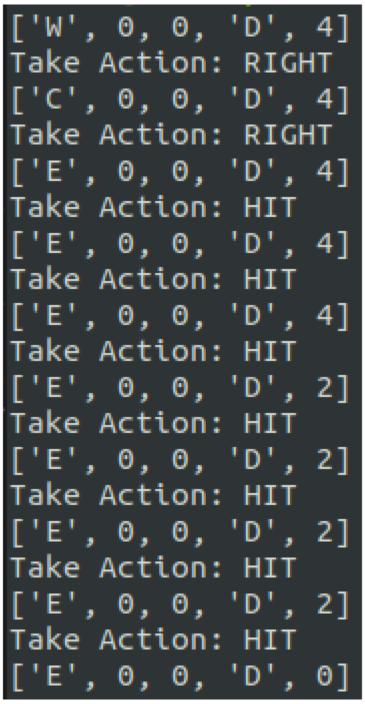
Rate of Convergence: 5.12

Iterations: 7 [Very Early convergence as later states are now weighed much less so they impact very little on the current result]

Different policies of actions selected by the different conditions given to us are given below, and these are consistent with the disscussion above.

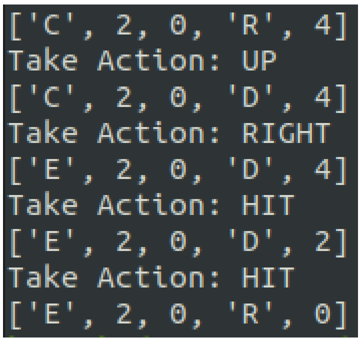
The Health Values in the following screenshots range between 0-4(inclusive). The health values are mapped like 0=>0, 1=>25, 2=>50, 3=>75, 4=>100

Task 1 => (a)



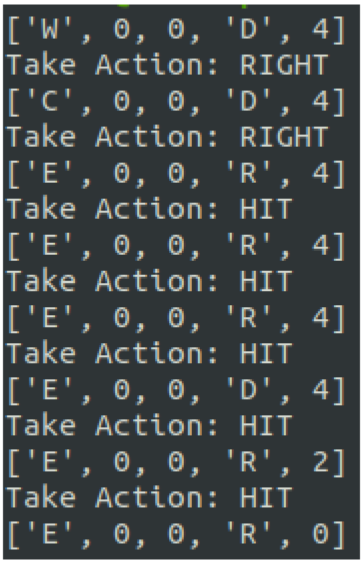
```
['W', 0, 0, 'D', 4]
Take Action: RIGHT
['C', 0, 0, 'D', 4]
Take Action: RIGHT
['E', 0, 0, 'D', 4]
Take Action: HIT
['E', 0, 0, 'D', 4]
Take Action: HIT
['E', 0, 0, 'D', 4]
Take Action: HIT
['E', 0, 0, 'D', 2]
Take Action: HIT
['E', 0, 0, 'D', 2]
Take Action: HIT
['E', 0, 0, 'D', 2]
Take Action: HIT
['E', 0, 0, 'D', 2]
Take Action: HIT
['E', 0, 0, 'D', 0]
```

(b)



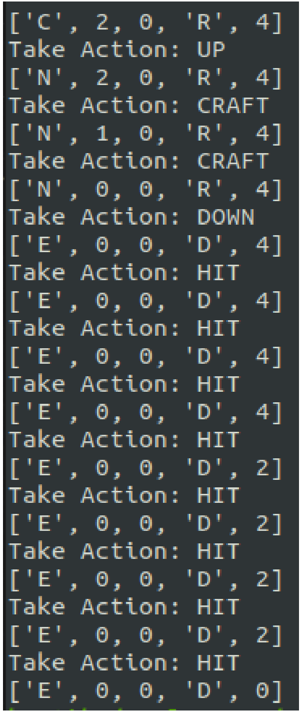
```
['C', 2, 0, 'R', 4]
Take Action: UP
['C', 2, 0, 'D', 4]
Take Action: RIGHT
['E', 2, 0, 'D', 4]
Take Action: HIT
['E', 2, 0, 'D', 2]
Take Action: HIT
['E', 2, 0, 'R', 0]
```

Task 2.1 => (a)



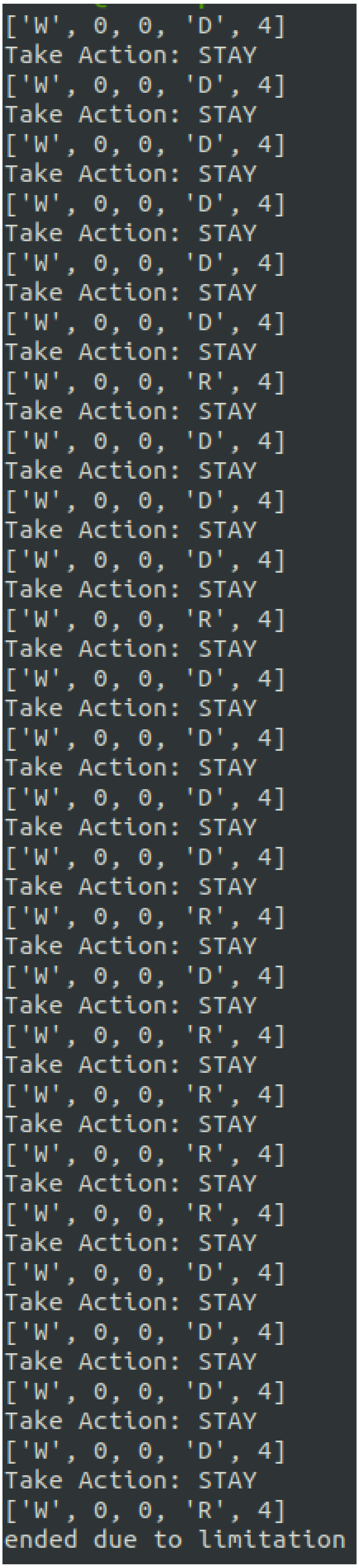
```
['W', 0, 0, 'D', 4]
Take Action: RIGHT
['C', 0, 0, 'D', 4]
Take Action: RIGHT
['E', 0, 0, 'R', 4]
Take Action: HIT
['E', 0, 0, 'R', 4]
Take Action: HIT
['E', 0, 0, 'R', 4]
Take Action: HIT
['E', 0, 0, 'D', 4]
Take Action: HIT
['E', 0, 0, 'R', 2]
Take Action: HIT
['E', 0, 0, 'R', 0]
```

(b)

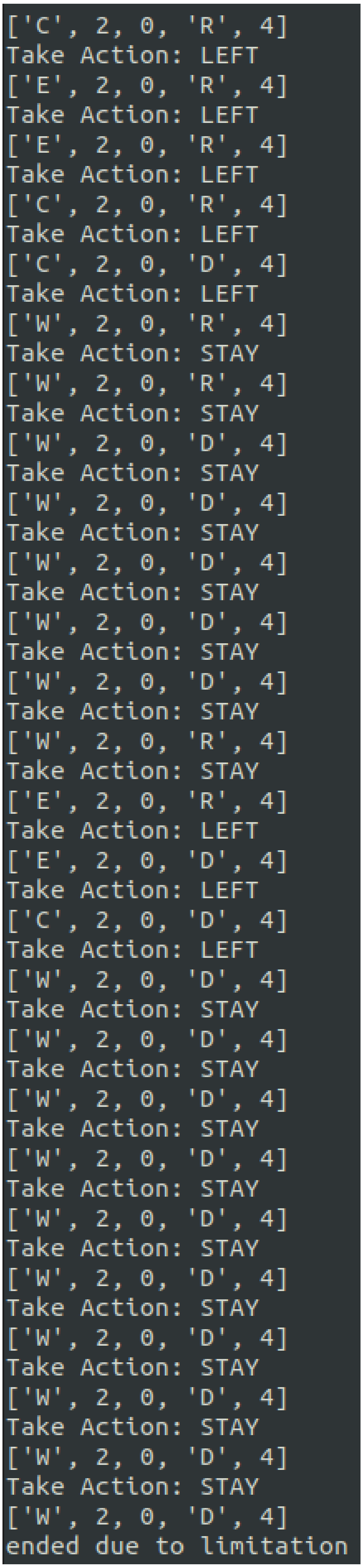


```
['C', 2, 0, 'R', 4]
Take Action: UP
['N', 2, 0, 'R', 4]
Take Action: CRAFT
['N', 1, 0, 'R', 4]
Take Action: CRAFT
['N', 0, 0, 'R', 4]
Take Action: DOWN
['E', 0, 0, 'D', 4]
Take Action: HIT
['E', 0, 0, 'D', 4]
Take Action: HIT
['E', 0, 0, 'D', 4]
Take Action: HIT
['E', 0, 0, 'D', 4]
Take Action: HIT
['E', 0, 0, 'D', 2]
Take Action: HIT
['E', 0, 0, 'D', 2]
Take Action: HIT
['E', 0, 0, 'D', 2]
Take Action: HIT
['E', 0, 0, 'D', 2]
Take Action: HIT
['E', 0, 0, 'D', 0]
```

Task 2.2=> (a) (Stuck) (b) (Stuck)

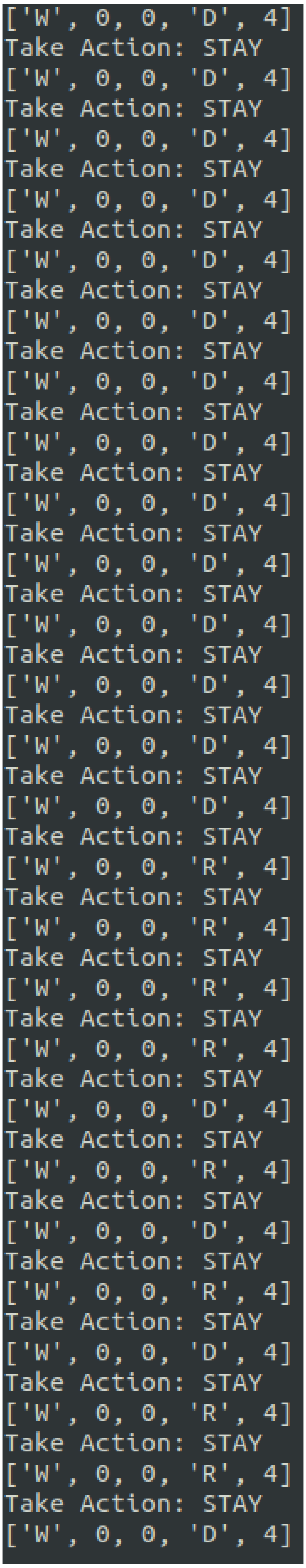


```
['W', 0, 0, 'D', 4]
Take Action: STAY
['W', 0, 0, 'D', 4]
Take Action: STAY
['W', 0, 0, 'D', 4]
Take Action: STAY
['W', 0, 0, 'D', 4]
Take Action: STAY
['W', 0, 0, 'D', 4]
Take Action: STAY
['W', 0, 0, 'D', 4]
Take Action: STAY
['W', 0, 0, 'R', 4]
Take Action: STAY
['W', 0, 0, 'D', 4]
Take Action: STAY
['W', 0, 0, 'D', 4]
Take Action: STAY
['W', 0, 0, 'D', 4]
Take Action: STAY
['W', 0, 0, 'R', 4]
Take Action: STAY
['W', 0, 0, 'D', 4]
Take Action: STAY
['W', 0, 0, 'D', 4]
Take Action: STAY
['W', 0, 0, 'D', 4]
Take Action: STAY
['W', 0, 0, 'D', 4]
Take Action: STAY
['W', 0, 0, 'R', 4]
Take Action: STAY
['W', 0, 0, 'D', 4]
Take Action: STAY
['W', 0, 0, 'R', 4]
Take Action: STAY
['W', 0, 0, 'R', 4]
Take Action: STAY
['W', 0, 0, 'R', 4]
Take Action: STAY
['W', 0, 0, 'R', 4]
Take Action: STAY
['W', 0, 0, 'D', 4]
Take Action: STAY
['W', 0, 0, 'D', 4]
Take Action: STAY
['W', 0, 0, 'D', 4]
Take Action: STAY
['W', 0, 0, 'D', 4]
Take Action: STAY
['W', 0, 0, 'R', 4]
ended due to limitation
```

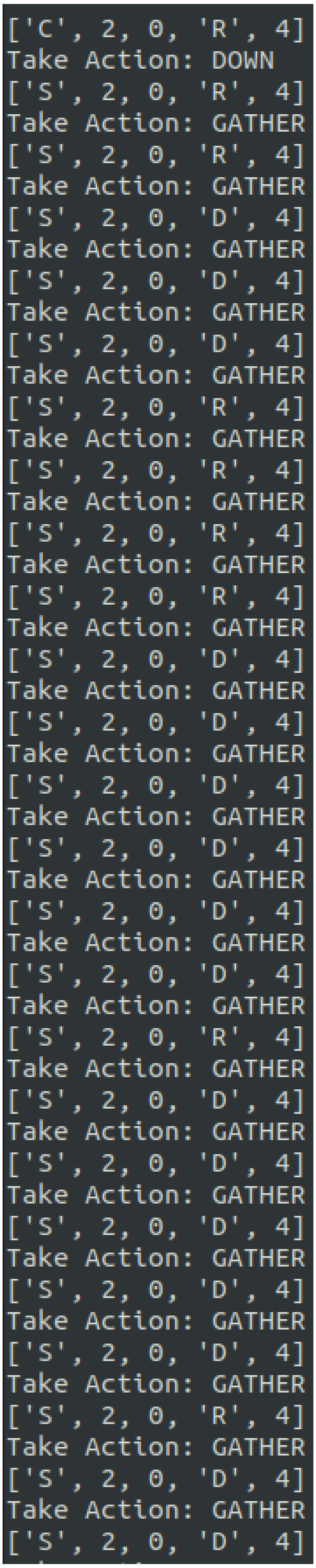


```
['C', 2, 0, 'R', 4]
Take Action: LEFT
['E', 2, 0, 'R', 4]
Take Action: LEFT
['E', 2, 0, 'R', 4]
Take Action: LEFT
['C', 2, 0, 'R', 4]
Take Action: LEFT
['C', 2, 0, 'D', 4]
Take Action: LEFT
['W', 2, 0, 'R', 4]
Take Action: STAY
['W', 2, 0, 'R', 4]
Take Action: STAY
['W', 2, 0, 'D', 4]
Take Action: STAY
['W', 2, 0, 'D', 4]
Take Action: STAY
['W', 2, 0, 'D', 4]
Take Action: STAY
['W', 2, 0, 'D', 4]
Take Action: STAY
['W', 2, 0, 'D', 4]
Take Action: STAY
['W', 2, 0, 'R', 4]
Take Action: STAY
['E', 2, 0, 'R', 4]
Take Action: LEFT
['E', 2, 0, 'D', 4]
Take Action: LEFT
['C', 2, 0, 'D', 4]
Take Action: LEFT
['W', 2, 0, 'D', 4]
Take Action: STAY
['W', 2, 0, 'D', 4]
Take Action: STAY
['W', 2, 0, 'D', 4]
Take Action: STAY
['W', 2, 0, 'D', 4]
Take Action: STAY
['W', 2, 0, 'D', 4]
Take Action: STAY
['W', 2, 0, 'D', 4]
Take Action: STAY
['W', 2, 0, 'D', 4]
Take Action: STAY
['W', 2, 0, 'D', 4]
Take Action: STAY
['W', 2, 0, 'D', 4]
Take Action: STAY
['W', 2, 0, 'D', 4]
ended due to limitation
```

Task 2.3 =>   (a)(Stuck)   (b)(Stuck)

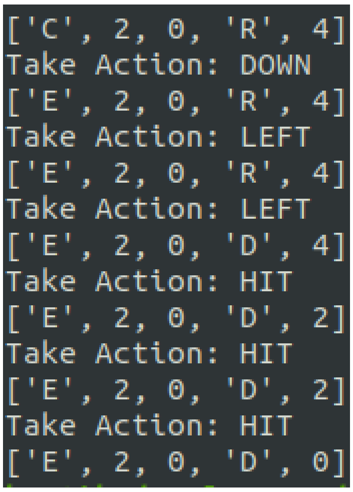


```
['W', 0, 0, 'D', 4]
Take Action: STAY
['W', 0, 0, 'D', 4]
Take Action: STAY
['W', 0, 0, 'D', 4]
Take Action: STAY
['W', 0, 0, 'D', 4]
Take Action: STAY
['W', 0, 0, 'D', 4]
Take Action: STAY
['W', 0, 0, 'D', 4]
Take Action: STAY
['W', 0, 0, 'D', 4]
Take Action: STAY
['W', 0, 0, 'D', 4]
Take Action: STAY
['W', 0, 0, 'D', 4]
Take Action: STAY
['W', 0, 0, 'D', 4]
Take Action: STAY
['W', 0, 0, 'D', 4]
Take Action: STAY
['W', 0, 0, 'D', 4]
Take Action: STAY
['W', 0, 0, 'D', 4]
Take Action: STAY
['W', 0, 0, 'D', 4]
Take Action: STAY
['W', 0, 0, 'R', 4]
Take Action: STAY
['W', 0, 0, 'R', 4]
Take Action: STAY
['W', 0, 0, 'R', 4]
Take Action: STAY
['W', 0, 0, 'R', 4]
Take Action: STAY
['W', 0, 0, 'D', 4]
Take Action: STAY
['W', 0, 0, 'R', 4]
Take Action: STAY
['W', 0, 0, 'D', 4]
Take Action: STAY
['W', 0, 0, 'R', 4]
Take Action: STAY
['W', 0, 0, 'D', 4]
Take Action: STAY
['W', 0, 0, 'R', 4]
Take Action: STAY
['W', 0, 0, 'R', 4]
Take Action: STAY
['W', 0, 0, 'D', 4]
```



```
['C', 2, 0, 'R', 4]
Take Action: DOWN
['S', 2, 0, 'R', 4]
Take Action: GATHER
['S', 2, 0, 'R', 4]
Take Action: GATHER
['S', 2, 0, 'D', 4]
Take Action: GATHER
['S', 2, 0, 'D', 4]
Take Action: GATHER
['S', 2, 0, 'D', 4]
Take Action: GATHER
['S', 2, 0, 'R', 4]
Take Action: GATHER
['S', 2, 0, 'R', 4]
Take Action: GATHER
['S', 2, 0, 'R', 4]
Take Action: GATHER
['S', 2, 0, 'R', 4]
Take Action: GATHER
['S', 2, 0, 'D', 4]
Take Action: GATHER
['S', 2, 0, 'D', 4]
Take Action: GATHER
['S', 2, 0, 'D', 4]
Take Action: GATHER
['S', 2, 0, 'D', 4]
Take Action: GATHER
['S', 2, 0, 'D', 4]
Take Action: GATHER
['S', 2, 0, 'D', 4]
Take Action: GATHER
['S', 2, 0, 'R', 4]
Take Action: GATHER
['S', 2, 0, 'D', 4]
Take Action: GATHER
['S', 2, 0, 'D', 4]
Take Action: GATHER
['S', 2, 0, 'D', 4]
Take Action: GATHER
['S', 2, 0, 'D', 4]
Take Action: GATHER
['S', 2, 0, 'D', 4]
Take Action: GATHER
['S', 2, 0, 'R', 4]
Take Action: GATHER
['S', 2, 0, 'D', 4]
Take Action: GATHER
['S', 2, 0, 'D', 4]
```

Task 2.3 => (b) [SOLVED]



```
['C', 2, 0, 'R', 4]
Take Action: DOWN
['E', 2, 0, 'R', 4]
Take Action: LEFT
['E', 2, 0, 'R', 4]
Take Action: LEFT
['E', 2, 0, 'D', 4]
Take Action: HIT
['E', 2, 0, 'D', 2]
Take Action: HIT
['E', 2, 0, 'D', 2]
Take Action: HIT
['E', 2, 0, 'D', 0]
```