WEEK-2

VACUUM CLEANER AGENT

CODE:

```
def vacuum_world():
    goal_state = {'A': '0', 'B': '0'}
    cost = 0

    location_input = input("Enter Location of Vacuum")
    status_input = input("Enter status of " + location_input)
    status_input_complement = input("Enter status of other room")
    print("Initial Location Condition" + str(goal_state))

    if location_input == 'A':
        print("Vacuum is placed in Location A")
        if status_input == '1':
            print("Location A is Dirty.")
            goal_state['A'] = '0'
            cost += 1
            print("Cost for CLEANING A " + str(cost))
            print("Location A has been Cleaned.")

            if status_input_complement == '1':
                print("Location B is Dirty.")
                print("Moving right to the Location B. ")
                cost += 1
                print("COST for moving RIGHT" + str(cost))
```

```
            goal_state['B'] = '0'
            cost += 1
            print("COST for SUCK " + str(cost))
            print("Location B has been Cleaned. ")
        else:
            print("No action" + str(cost))
            print("Location B is already clean.")

    if status_input == '0':
        print("Location A is already clean ")
        if status_input_complement == '1':# if B is Dirty
            print("Location B is Dirty.")
            print("Moving RIGHT to the Location B. ")
            cost += 1
            print("COST for moving RIGHT " + str(cost))
            goal_state['B'] = '0'
            cost += 1
            print("Cost for SUCK" + str(cost))
            print("Location B has been Cleaned. ")
        else:
            print("No action " + str(cost))
            print(cost)
            print("Location B is already clean.")

else:
    print("Vacuum is placed in location B")
    if status_input == '1':
```

```
    print("Location B is Dirty.")
    goal_state['B'] = '0'
    cost += 1
    print("COST for CLEANING " + str(cost))
    print("Location B has been Cleaned.")

    if status_input_complement == '1':
        print("Location A is Dirty.")
        print("Moving LEFT to the Location A. ")
        cost += 1
        print("COST for moving LEFT" + str(cost))
        goal_state['A'] = '0'
        cost += 1
        print("COST for SUCK " + str(cost))
        print("Location A has been Cleaned.")

else:
    print(cost)
    print("Location B is already clean.")

    if status_input_complement == '1':
        print("Location A is Dirty.")
        print("Moving LEFT to the Location A. ")
        cost += 1
        print("COST for moving LEFT " + str(cost))
        goal_state['A'] = '0'
        cost += 1
```

```
            print("Cost for SUCK " + str(cost))
            print("Location A has been Cleaned. ")
        else:
            print("No action " + str(cost))
            print("Location A is already clean.")


    print("GOAL STATE: ")
    print(goal_state)
    print("Performance Measurement: " + str(cost))


vacuum_world()
```

OUTPUT:

```
Enter Location of VacuumA
Enter status of A1
Enter status of other room1
Initial Location Condition{'A': '0', 'B': '0'}
Vacuum is placed in Location A
Location A is Dirty.
Cost for CLEANING A 1
Location A has been Cleaned.
Location B is Dirty.
Moving right to the Location B.
COST for moving RIGHT2
COST for SUCK 3
Location B has been Cleaned.
GOAL STATE:
{'A': '0', 'B': '0'}
Performance Measurement: 3
```

```
Enter Location of Vacuum A
Enter status of  A 0
Enter status of other room 0
Initial Location Condition{'A': '0', 'B': '0'}
Vacuum is placed in location B
0
Location B is already clean.
No action 0
Location A is already clean.
GOAL STATE:
{'A': '0', 'B': '0'}
Performance Measurement: 0
```

Analysis:

1. **Input and Initialization:**
    - Takes user input for the initial location (**location_input**) and room status (**status_input** and **status_input_complement**).
    - Initializes the goal state (**goal_state**) and cost (**cost**).
2. **Cleaning Process:**
    - Based on the user input, the code cleans the dirty room in the initial location and moves to the other location if needed.
    - Actions include updating cleanliness status, calculating costs, and printing messages.
3. **Goal State and Cost Tracking:**
    - Prints the final cleanliness status (**goal_state**) as the goal state.
    - Displays the total cost (**cost**) as a performance measurement.
4. **Location-Aware Cleaning:**
    - Adapts cleaning actions according to the initial location of the vacuum (A or B).
    - Cleans the room, moves to the other room if necessary, and updates cleanliness status.

5. **Simplified Simulation:**
    - Provides a basic simulation of a vacuum cleaner's cleaning process in a two-room environment.
    - Assumes valid user input and lacks extensive error handling for simplicity.