2. Implement vacuum cleaner agent

```
def vacuum_world():
    # Initializing goal_state
    # 0 indicates Clean and 1 indicates Dirty
    goal_state = {'A': '0', 'B': '0'}
    cost = 0


    location_input = input("Enter Location of Vacuum (A or B): ").strip().upper()
# User input of location
    status_input = input(f"Enter status of {location_input} (0 for Clean, 1 for
Dirty): ").strip()
    status_input_complement = input("Enter status of other room (0 for Clean, 1 for
Dirty): ").strip()

    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'  # Clean A
            cost += 1  # Cost for sucking
            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 Location B.")
                cost += 1  # Cost for moving right
                print("Cost for moving RIGHT: " + str(cost))

                goal_state['B'] = '0'  # Clean B
                cost += 1  # Cost for sucking
                print("Cost for SUCK: " + str(cost))
                print("Location B has been Cleaned.")
            else:
                print("Location B is already clean.")
        else:
            print("Location A is already clean.")
            if status_input_complement == '1':
                print("Location B is Dirty.")
                print("Moving RIGHT to Location B.")
                cost += 1  # Cost for moving right
                print("Cost for moving RIGHT: " + str(cost))
```

```
            goal_state['B'] = '0'  # Clean B
            cost += 1  # Cost for sucking
            print("Cost for SUCK: " + str(cost))
            print("Location B has been Cleaned.")
        else:
            print("Location B is already clean.")

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

        if status_input_complement == '1':
            print("Location A is Dirty.")
            print("Moving LEFT to Location A.")
            cost += 1  # Cost for moving left
            print("Cost for moving LEFT: " + str(cost))

            goal_state['A'] = '0'  # Clean A
            cost += 1  # Cost for sucking
            print("Cost for SUCK: " + str(cost))
            print("Location A has been Cleaned.")
        else:
            print("Location A is already clean.")
    else:
        print("Location B is already clean.")
        if status_input_complement == '1':
            print("Location A is Dirty.")
            print("Moving LEFT to Location A.")
            cost += 1  # Cost for moving left
            print("Cost for moving LEFT: " + str(cost))

            goal_state['A'] = '0'  # Clean A
            cost += 1  # Cost for sucking
            print("Cost for SUCK: " + str(cost))
            print("Location A has been Cleaned.")
        else:
            print("Location A is already clean.")

# Done cleaning
print("GOAL STATE: ")
print(goal_state)
print("Performance Measurement: " + str(cost))
```

```
# Output
vacuum_world()
print("-----------")
print("Varsha P(1BM22CS320)")
```

Output

```
Enter Location of Vacuum (A or B): A
Enter status of A (0 for Clean, 1 for Dirty): 1
Enter status of other room (0 for Clean, 1 for Dirty): 1
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 Location B.
Cost for moving RIGHT: 2
Cost for SUCK: 3
Location B has been Cleaned.
GOAL STATE:
{'A': '0', 'B': '0'}
Performance Measurement: 3
-----------
Varsha P(1BM22CS320)
```

LAB-2

Vacuum World Algorithm

```
def vacuum_world():
    goal_state = {'A': '0', 'B': '0'}
    cost = 0
location_input = input("Enter location of Vacuum (A or B): ").strip().upper()
status_input = input(f"Enter status of {location_input} (0 for clean, 1 for dirty): ")
other_location = 'B' if location_input == 'A' else 'A'
status_input_complement = input(f"Enter status of {other_location} (0 for clean, 1 for dirty): ").strip()

print("Initial Location Condition:", goal_state)
def clean(location):
    nonlocal cost
    print(f"Location {location} is Dirty")
    goal_state[location] = '0'
    cost += 1
    print(f"Cost for cleaning {location}: {cost}")
    print(f"Location {location} has been cleaned.")
if status_input == '1':
    clean(location_input)
if status_input_complement == '1':
    print(f"Moving Right to Location {other_location}")
    cost += 1
    print(f"Cost for moving : {cost}")
    clean(other_location)
else:
    print(f"No action. Location {other_location} is already clean.")
```