Date 1-10-24

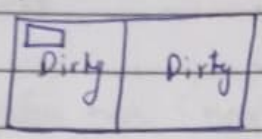
Date __/__/__
Page _____

LAB-2

Vacuum Cleaner Problem

Two rooms

Step 1:



Initialize two states for a room being location and status. Location is the room number, either 1 or 2 and the status is either clean or dirty

~~room=[1,2]~~

~~Vacuum_cleaner (room)~~

~~room[1]="Dirty"~~

~~room[2]="Dirty"~~

~~room[1,2]~~ room = [1, 2]

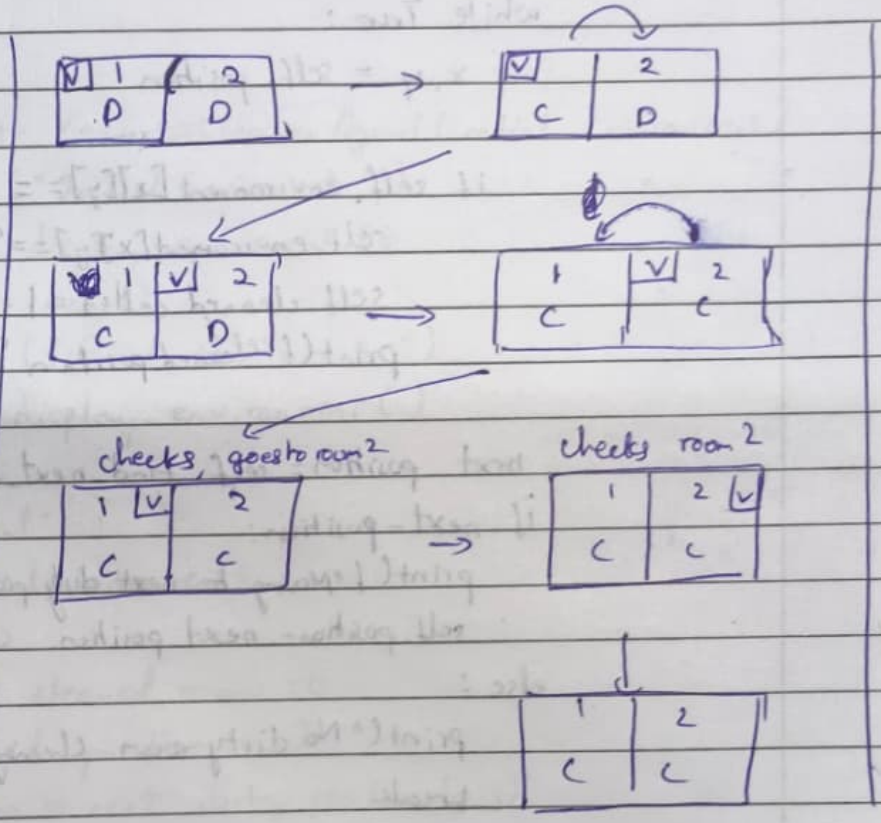
```
clean_room (room)
{
    room_status = ["Dirty", "Clean"]
    room.1 = "Dirty"
    room.2 = "Dirty"
}

Vacuum_cleaner (room)
{
    initial_room = room 1
    if (room.1 = "Dirty")
    { cleans the room, set status to clean
    since room is clean, it goes left.
    if (room.2 = "Dirty")
        cleans the room, set status to clean
```

Date __/__/__
Page _____

since room is clean, it goes right.

if (room 1 is clean)
state remains same, it goes left

if (room 2 is clean)
state remains same, it turns off.



(1, dirty) ⊖

(1, clean) → (2, dirty) and stops
move left

(1, clean) (2, clean) (1, clean)

Date __/__/__
Page _____

```
class VacuumCleaner:
    def __init__(self, environment):
        self.environment = environment
        self.cleaned_cells = 0
        self.position = (0,0)

    def clean(self):
        while True:
            x, y = self.position

            if self.environment[x][y] == 'D':
                self.environment[x][y] == 'D':
                self.cleaned_cells += 1
                print(f"cleaned position {self.position}")

            next_position = self.find_next_dirty()
            if next_position:
                print(f"Moving to next dirty position {next_position}")
                self.position = next_position
            else:
                print("No dirty room. Cleaning Complete")
                break

    def find_next_dirty(self):

        for i in range(len(self.environment)):
            for j in range(len(self.environment[i])):
                if self.environment[i][j] == 'D':
                    return (i,j)
        return None
```

```
def display_environment(self):
    for row in self.environment:
        print(" ".join(row))
    print(f"Total cleaned cells: {self.cleaned_cells}")

initial_environment = [
    ['D', 'D'],
    [illegible]
]

agent = VacuumCleanerAgent(initial_environment)
print("Initial Environment")
agent.display_environment()
agent.clean()
print("Final Environment")
agent.display_environment()
```

Output:

Initial environment:

D D

Total cleaned rooms: 0

Cleaned position (0, 0)

Moving to next dirty position (0, 1)

Cleaned position (0, 1)

No more dirty cells

Final environment:

C C

Total cleaned cells: 2

Date __/__/__
Page

Initial environment:
D D
D D
Total Cleaned cells : 0
cleaned position (0,0)
Moving to next dirty position (0,1)
Cleaned position (0,1)
Moving to next dirty position (1,0)
Cleaned position (1,1)
No more dirty rooms

Final environment:
C C
C C