Programs

Vaccum Cleaner

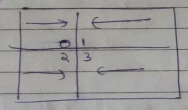
~~def clean():~~

* Algorithm:-

① We have a 2D list called 'floor'

We take input for no of rows & columns of the floor grid.

The user then inputs initial state of each cell in this grid (1 for dirty & 0 for clean)
→ You do this for each room

② 

Total of 4 rooms

0 & 2 room - Cleaner moves L to R

1 & 3 room - Cleaner moves R to L

③ In each room, ~~the~~ cleaner must check each cell in the grid.

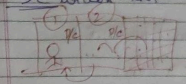
If ~~floor~~ cell is dirty, it cleans it by setting the value to 1.

floor(i)(j) = 0

④ After cleaning each cell, the current state of ~~floor~~ room is displayed.

To show ~~the~~ where the cleaner is at

><



① User inputs state of each room
↓
this is stored in a list room state

② Clean room → function that takes room state as argument

If clean room 0 to 3 and returns to 0th room

③ If states of room = 1 → dirty
clean it and makes the state = 0

④ After returning to 0th room it gives a completed message

Code:-

```
def clean_rooms(rooms):
    for i in range(len(rooms)):
        clean_room(rooms, i)

clean_room(room, 0)

def clean_room(rooms, i):
    if rooms[i] == 1:
        print(f"Cleaning room {i}")
        room[i] = 0
    else:
        print(f"Room {i} is already clean")

~~def main():~~

rooms = 4
room_state = []

print("Initial state for each room")
for room in range(rooms):
    state = int(input(f"Room {room} state (0/1): "))
    room_state.append(state)

clean_rooms(room_state)

print("All rooms have been present")
```

Output:-

```
Enter initial state for 4 rooms
Room 0 state: 0
Room 1 state: 1
Room 2 state: 1
Room 3 state: 0

Room 0 is already clean
Cleaning Room 1
Cleaning Room 2
Room 3 is already clean
Room 0 is already clean

All rooms have been present
```

```
0 indicates clean and 1 indicates dirty
harshala Rani 1BM21CS074
Enter Location of VacuumA
Enter status of A1
Enter status of other room1
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
```