Week - 02

01/10/2024

Date __/__/__
Page _____

Algorithm for Vaccum Cleaner

① Firstly considering two rooms for the cleaning.

② check and make sure that the vaccum cleaner should clean the two rooms and returns to the initial state again after the completion of two room

③ consider location of the room and states of the room

0 → represents clean

1 → represents dirty

vaccum cleaner can move left, Right, up and down because it will working in 2-d grid

```
def isdirty():
    return self.rooms[self.position] == 1

def isclean():
    if self.isdirty():
        self.rooms[self.position] = 0
        self.cleaned_rooms += 1

def move():
    self.position = 1 - self.position

def run(steps):
    for step in rang(steps):
        clean()
        move()
rooms = [1, 0]
```

Date __/__/__
Page _____

Percept sequence

check : RoomA , Dirty
Action : clean RoomA & move

check : RoomB , Dirty
Action : clean RoomB & move

I :→ (Room 1, Dirty)
II :→ (Room 2, clean)
III :→ (Room 1, clean)
IV :→ (Room 2, clean)
V :→ (Room 1, clean)
VI :→ (Room 2, clean)

Code :-

```
class vaccum cleaner :
    def __init__(self, rooms, start_position):

        self.rooms = rooms
        self.position = start_position
        self.cleaned_rooms = 0
        self.Percept_sequence = [ ]

    def is_dirty(self):
        return self.rooms[self.position] == 1

    def clean(self):
        if self.is_dirty():
            print(f"cleaning room {self.position
                                    + 1}")
```

Date __/__/__
Page _____

```
def move(self):
    if self.position<len(self.rooms)-1:
        self.position += 1
    else
        self.position = 0
    print(f"moved to room {self.position+1}")

def perceive(self):
    room_state = "Dirty" if self.is_dirty()
        else "clean"
    percept = (f"Room {self.position+1}",
                room_state)
    self.percept_sequence.append(percept)
    print(f"perception: {percept}")

def run(self, steps):
    for step in range(steps):
        print(f"step {step+1}:")
        self.perceive()
        self.clean()
        self.move()

print(f"Rooms states: {self.rooms}\n")
print(f"Total cleaned rooms: {self.cleaned_
                    room}")
print("Percept sequence:", self.percept_sequence)
rooms = [1, 0, 1, 1]
vaccum = vaccumcleaner(rooms, start_position=0)
vaccum.run(steps=8)
```

Date __/__/__
Page _____

Sample Output :-

step 1 :
Perception : ('Room 1', 'Dirty')
cleaning Room 1
move to Room 2
Rooms states = [0, 0, 1, 1]
step 2 :
Perception : ('Room 2', 'clean')
moved to Room 3
Rooms states = [0, 0, 1, 1]

step 3 :
Perception : ('Room 3', 'Dirty')
cleaning Room 3
moved to room 4
Rooms states = [0, 0, 0, 1]

Step 4 :
Perception : ('Room 4', 'Dirty')
cleaning room 4
move 4 to room 1
Room states = [0, 0, 0, 0]

Step 5 :
Perception : ('Room 1', 'clean')
moved to room 2
Rooms states : [0, 0, 0, 0]

step 6 :-
Perception : ('Room 2', 'clean')
moved to room 3
Rooms states : [0, 0, 0, 0]

Date __/__/__
Page _____

step 7 :
Perception : ('Room 3', 'clean')
moved to room 4
Room status : [0, 0, 0, 0]

step 8 :
Perception : ('Room 4', 'clean')
moved to room 1
Rooms status : [0, 0, 0, 0]

Total cleaned rooms : 3
Percept sequence :
[(Room 1, Dirty)
(Room 2, clean)
(Room 3, Dirty)
(Room 4, Dirty)
(Room 1, clean)
(Room 2, clean)
(Room 3, clean)
(Room 4, clean)]