**NAME:** NIHA

**USN:** 1BM18CS060

**DATE:** 07-11-2020

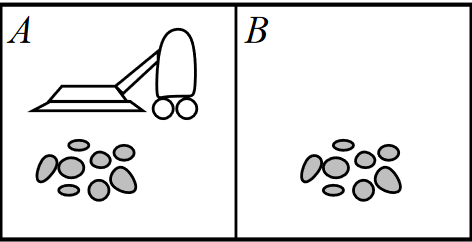
**LAB 5**: Implement vacuum cleaner agent.

**Problem Statement**

**Vacuum Cleaner Agent:**

It is a goal-based agent, and the goal of this agent, which is the vacuum cleaner, is to clean up the whole area. The vacuum cleaner can perform the following functions: move left, move right, move forward, move backward and to suck dust.

Precepts: location and contents, e.g., [A, Dirty]

Actions: Left, Right, Up, Down, Suck, NoOp

**Explanation:**

**Percepts**: The percept sequence consists of the history of percepts the vacuum cleaner senses, and the agent function maps each percept of the sequence to an action. The agent function for this vacuum cleaning agent is:

Agent function = {([A: Clean], Right), ([A: Dirty], Clean), ([B: Clean], Left), ([B: Dirty], Clean), ([A: Clean, B: Clean], Stop), ([A: Clean, B: Dirty], Clean), …}

**Actions**: There are five actions available: go forward, turn right by 90°, turn left by 90°, suck up dirt, and turn off.

**Environment**: The environment consists of a grid of squares. Some squares contain obstacles (walls and furniture) and other squares are open space. Some of the open squares contain dirt. Each "go forward" action moves one square unless there is an obstacle in that square, in which case the agent stays where it is, but the touch sensor goes on. A "suck up dirt" action always cleans up the dirt. A "turn off" command ends the simulation.

**Pseudocode:**

function Reflex-Vacuum-Agent( [location, status]) returns an action

If status = Dirty then return Suck

else if location =A then return Right

else if location = B then return Left

Since here there are only two rooms, the possibilities of agent to move are only left and right.

If there are more than two rooms then the possibilities will be left, right, up, and down.

**Implementing vaccum cleaner agent**:

Based on this information we implement vaccum cleaner agent for any number of rooms. Code follow below:

1. To clean a room

def clean(floor):

1. Declare a grid of size m x n
2. Here 0 represents the clean state of room and 1 represents the dirty state of room
3. Traverse through the grid depending on the value i.e., clean (0) or dirty (1), if the state is 1 then change state to 0 else no action and continue to check for states of other rooms.
4. Depending on the room location the agent moves left, right, up, and down.

2. To print the states of room in each move

def print\_floor(floor, row, col):

1. Printing the grid every time we move another room to check the state of room.
2. Once the states of all rooms are 0 it shows that all rooms are cleaned.