To run this vacuum cleaner simulation, you'll need a Python environment with the random module (which is part of the standard library) installed. The simulation operates in a 2D grid environment where each cell is either clean or dirty, represented by a Room class. The room's grid is randomly initialized, with some cells marked dirty (value 1) and others clean (value 0). A VacuumCleaner agent is randomly placed in the room, and it can move in four directions: up, down, left, or right. The vacuum cleaner uses a random\_walk function to choose a random direction and cleans each cell it encounters. The program continues this process until all cells in the room are clean, and after each move and clean action, the room's state is displayed. The core logic revolves around the vacuum cleaner making random moves, checking the cleanliness of its current location, and cleaning the cell if it is dirty.  
  
1. Room class methods:

* \_\_init\_\_(self, width, height):  
  Initializes the room as a grid of the specified width and height. Each cell is randomly marked as clean (0) or dirty (1).
* display(self):  
  Prints the current status of the room's grid, showing which cells are clean and which are dirty.
* is\_dirty(self, x, y):  
  Checks if the cell at position (x, y) is dirty (returns True if dirty, False if clean).
* clean\_cell(self, x, y):  
  Cleans the cell at position (x, y) if it is dirty, changing its value from 1 to 0. It prints a message indicating whether the cell was cleaned or was already clean.
* is\_all\_clean(self):  
  Checks if all the cells in the room are clean. It returns True if every cell in the grid is clean (0), and False otherwise.

### 2. **VacuumCleaner class methods:**

* \_\_init\_\_(self, room):  
  Initializes the vacuum cleaner, randomly placing it in a room. The vacuum's initial position is set to a random (x, y) coordinate within the room's grid.
* move(self, direction):  
  Moves the vacuum cleaner in a specified direction ('up', 'down', 'left', or 'right') while ensuring it doesn’t move outside the room's boundaries. It prints the vacuum's new position after moving.
* clean(self):  
  Cleans the current cell where the vacuum cleaner is located by calling the room's clean\_cell method.
* random\_walk(self):  
  Chooses and returns a random direction ('up', 'down', 'left', or 'right') for the vacuum cleaner to move.

### 3. **Main simulation function:**

* run\_vacuum\_cleaner():  
  The main function that creates a room and a vacuum cleaner, then runs the simulation. It continues to clean and move the vacuum around randomly until all the cells in the room are clean, printing the state of the room after each step. It ends by printing a message that the room is fully cleaned.

### 4. **Script execution check:**

* if \_\_name\_\_ == "\_\_main\_\_":  
  Ensures that the simulation runs only if the script is executed directly, and not if it’s imported as a module. This calls the run\_vacuum\_cleaner() function to start the simulation.