335-05 Project 1 Cella Ant #12 – Big-O Analysis

For this program, N is the number of moves the ant performs. The program runs the function update N number of times. Inside update(), there are turn_ant(), change_cell_state(), move_forward_ant(), an increment, and an if statement calling update() if true.

The update() calls turn_ant() which has assignment operators O(1), if statements with ternary operators inside O(1), fill_cell() which will be mentioned next, and draw_ant() which will be mentioned later. In fill_cell(), there are assignment operators O(1), switch statement with assignment operators O(1), and fillRect() O(1). The time complexity of fill_cell() is O(1). In draw_ant(), there are assignment operators O(1) and a switch statement calling draw_triangle() or console.log(), draw_triangle() should be O(1) and console.log() is O(1). The time complexity of draw_ant() is O(1), so turn_ant() should be O(1).

The update() calls change_cell_state() next which has assignment operators O(1), ternary operator with assignment operator and increment operator which is both O(1), and fill_cell() O(1). The time complexity of change_cell_state() is O(1).

The update() calls move_forward_ant() which has assignment operators O(1), switch statement with increment operator O(1) and console.log() O(1), and draw_ant() O(1). The time complexity of move_forward_ant() is O(1).

Next is the increment operator which is O(1). The if statement runs update() again if true. The program recursively calls update() N times. The time complexity of the entire program is O(N).