Team GAC
Graciela Orozco
Arely De Leon
Christopher Bui

CPSC 335-02 Team GAC Project #1: Langston's Ant Project Complexity Analysis

Overall Complexity

The overall time complexity of the program runs at O(N) time. Our functions included, are mostly O(1) alongside traversing the array of O(N). In conclusion,
$O(1) + O(1) + O(1) + O(1)\ O(1) + O(1) + O(N) \approx O(N)$

draw_triangle(), draw_square(), and draw_grid()

The functions draw_triangle, draw_square, and draw_grid and take O(1) time to execute. Each command within the function is immediate since they consist of basic operations such as moving and filling in. These commands all take O(1) space since they store x and y values temporarily to indicate where the cursor should move.

get_triangle_coordinates() and get_square_coordinates()

The set_triangle_coordinates and get_square_coordinates functions are similar in time and space complexity as they just reuse the draw_triangle and draw_square functions but add static numbers known at compile time. They are O(1) in both space and time complexity since the input size is based on the integer points inputted.

draw_array()

Given our array, check the squares from zero to forty-one, and once again in a nested for loop. Afterward, check the value of the square and print the square to that value. Therefore, our draw_array function adds O(41) + O(41) + O(1). Our resulting time complexity would be O(1).

traverse_array()

The traverse_array function runs in O(N) time. Within this function, it initializes an array to 0, changes values dependent on our marker, and prints out our results. When creating the array, we have set our 'i' and 'j' values to forty-one. Therefore, it is O(1) time to create an array. Afterwards, we set our marker at a point within our array. After N iterations, it would change it value of its array, position on the array, and direction it is facing. The time of O(N) + O(1) + O(1) +O (1). Our resulting time complexity would be O(N).