**Recursive Implementaion**

// Algorithm: recursive\_candles

// Input: Array of candles, number of candles (n), current height, maximum height

Algorithm recursive\_candles(candles: array of integers, n: integer, current\_height: integer, max\_height: integer) returns MaxHeightCount

{

// Check if there are no more candles

if n = 0 then

{

// Create a MaxHeightCount structure with the current max height and count 0

result := {.height = max\_height, .count = 0};

return result; // Return the result

}

// Initialize result with the current height and count 0

result := {.height = current\_height, .count = 0};

count := 0; // Initialize count to 0

// Check if the first candle height equals the current height

if candles[0] = current\_height then

{

count := count + 1; // Increment count

}

// Get the pointer to the rest of the candles

rest\_of\_candles := candles + 1;

// Recursively call the function with the rest of the candles

recursive\_result := recursive\_candles(rest\_of\_candles, n - 1, current\_height, max\_height);

// Update the result count if the sum of current count and recursive count is greater

if count + recursive\_result.count > result.count then

{

result.count := count + recursive\_result.count;

}

// Update the result height and count if the recursive result height is greater

if recursive\_result.height > result.height then

{

result.height := recursive\_result.height;

result.count := recursive\_result.count;

}

return result; // Return the result

}

Algorithm recursive\_birthdayCandles(candles: array of integers, n: integer) returns integer

{

// Check if there are no candles

if n <= 0 then

{

return 0; // Return 0 if there are no candles

}

// Call the recursive\_candles function to find max height count

max\_height\_count := recursive\_candles(candles, n, candles[0], 0);

return max\_height\_count.count; // Return the count of candles with max height

}