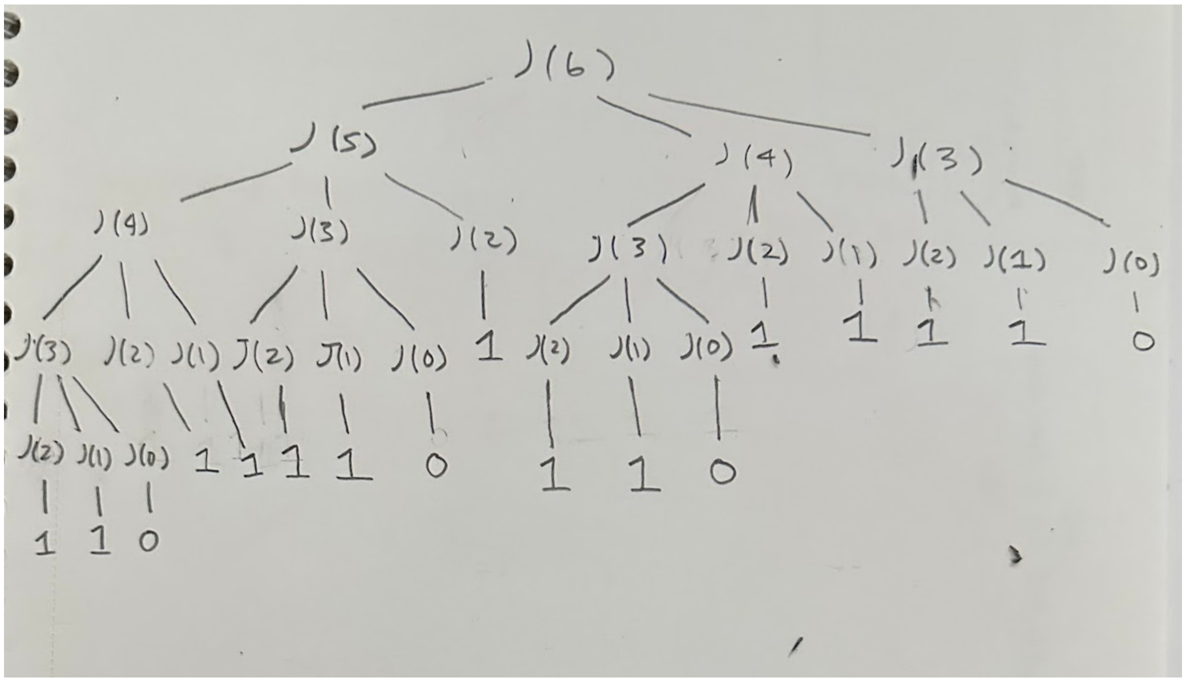
CSC 340 Assignment 3
Alex Kidwai, Ryan Zhang

Part 1:

2.


J(6)
J(5)
J(4)
J(3)
J(4)
J(3)
J(2)
J(3)
J(2)
J(1)
J(2)
J(1)
J(0)
J(3) J(2) J(1) J(2) J(1) J(0) 1 J(2) J(1) J(0) 1 1 1 1 0
J(2) J(1) J(0) 1 1 1 1 0 1 1 0
1 1 0

3. a.

Memoization is the practice of storing the results of large function calls to be reused later.
https://www.geeksforgeeks.org/what-is-memoization-a-complete-tutorial/

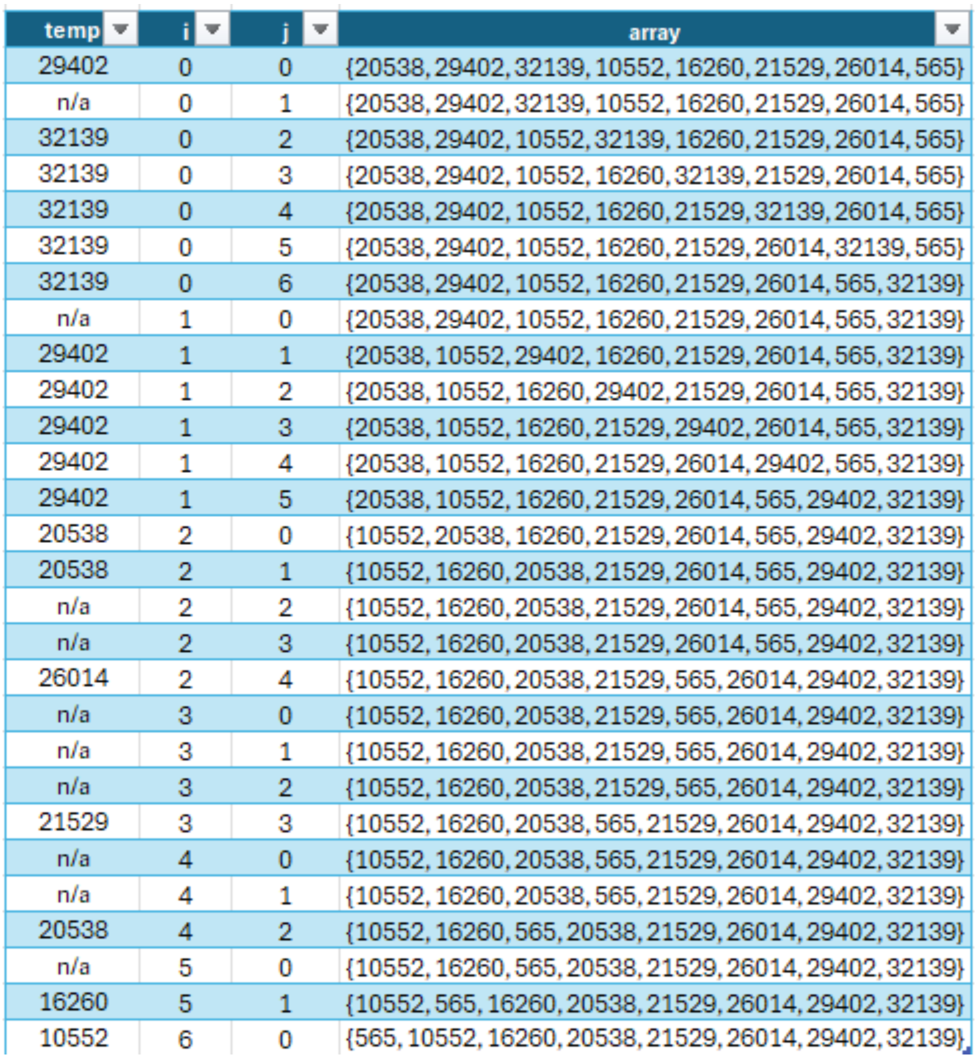
b.

- I would initialize an integer array of size 100 inside the function.
- The result of the function grows so quickly that n would reach the max integer size for long very quickly, so the size of the array doesn't have to be large.
- The indices can represent the value for n, so the result of an input of 2 would be stored at index 2.
- Before a recursive call for n > 2, the function checks if the value at the value of n is greater than zero. If it is, it uses the value in the array instead of a recursive call.

Part 2:

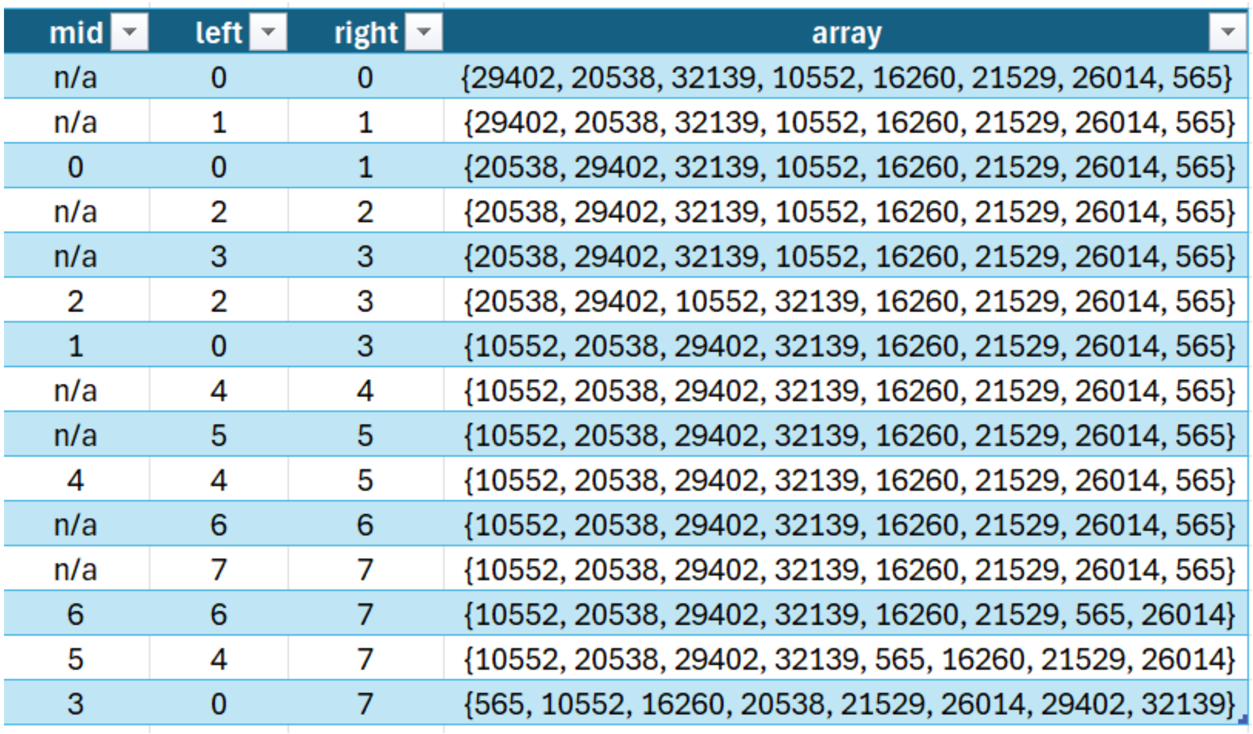
Array: {29402, 20538, 32139, 10552, 16260, 21529, 26014, 565}

Bubble sort:



temp	i	j	array
29402	0	0	{20538, 29402, 32139, 10552, 16260, 21529, 26014, 565}
n/a	0	1	{20538, 29402, 32139, 10552, 16260, 21529, 26014, 565}
32139	0	2	{20538, 29402, 10552, 32139, 16260, 21529, 26014, 565}
32139	0	3	{20538, 29402, 10552, 16260, 32139, 21529, 26014, 565}
32139	0	4	{20538, 29402, 10552, 16260, 21529, 32139, 26014, 565}
32139	0	5	{20538, 29402, 10552, 16260, 21529, 26014, 32139, 565}
32139	0	6	{20538, 29402, 10552, 16260, 21529, 26014, 565, 32139}
n/a	1	0	{20538, 29402, 10552, 16260, 21529, 26014, 565, 32139}
29402	1	1	{20538, 10552, 29402, 16260, 21529, 26014, 565, 32139}
29402	1	2	{20538, 10552, 16260, 29402, 21529, 26014, 565, 32139}
29402	1	3	{20538, 10552, 16260, 21529, 29402, 26014, 565, 32139}
29402	1	4	{20538, 10552, 16260, 21529, 26014, 29402, 565, 32139}
29402	1	5	{20538, 10552, 16260, 21529, 26014, 565, 29402, 32139}
20538	2	0	{10552, 20538, 16260, 21529, 26014, 565, 29402, 32139}
20538	2	1	{10552, 16260, 20538, 21529, 26014, 565, 29402, 32139}
n/a	2	2	{10552, 16260, 20538, 21529, 26014, 565, 29402, 32139}
n/a	2	3	{10552, 16260, 20538, 21529, 26014, 565, 29402, 32139}
26014	2	4	{10552, 16260, 20538, 21529, 565, 26014, 29402, 32139}
n/a	3	0	{10552, 16260, 20538, 21529, 565, 26014, 29402, 32139}
n/a	3	1	{10552, 16260, 20538, 21529, 565, 26014, 29402, 32139}
n/a	3	2	{10552, 16260, 20538, 21529, 565, 26014, 29402, 32139}
21529	3	3	{10552, 16260, 20538, 565, 21529, 26014, 29402, 32139}
n/a	4	0	{10552, 16260, 20538, 565, 21529, 26014, 29402, 32139}
n/a	4	1	{10552, 16260, 20538, 565, 21529, 26014, 29402, 32139}
20538	4	2	{10552, 16260, 565, 20538, 21529, 26014, 29402, 32139}
n/a	5	0	{10552, 16260, 565, 20538, 21529, 26014, 29402, 32139}
16260	5	1	{10552, 565, 16260, 20538, 21529, 26014, 29402, 32139}
10552	6	0	{565, 10552, 16260, 20538, 21529, 26014, 29402, 32139}

Merge Sort:



mid	left	right	array
n/a	0	0	{29402, 20538, 32139, 10552, 16260, 21529, 26014, 565}
n/a	1	1	{29402, 20538, 32139, 10552, 16260, 21529, 26014, 565}
0	0	1	{20538, 29402, 32139, 10552, 16260, 21529, 26014, 565}
n/a	2	2	{20538, 29402, 32139, 10552, 16260, 21529, 26014, 565}
n/a	3	3	{20538, 29402, 32139, 10552, 16260, 21529, 26014, 565}
2	2	3	{20538, 29402, 10552, 32139, 16260, 21529, 26014, 565}
1	0	3	{10552, 20538, 29402, 32139, 16260, 21529, 26014, 565}
n/a	4	4	{10552, 20538, 29402, 32139, 16260, 21529, 26014, 565}
n/a	5	5	{10552, 20538, 29402, 32139, 16260, 21529, 26014, 565}
4	4	5	{10552, 20538, 29402, 32139, 16260, 21529, 26014, 565}
n/a	6	6	{10552, 20538, 29402, 32139, 16260, 21529, 26014, 565}
n/a	7	7	{10552, 20538, 29402, 32139, 16260, 21529, 26014, 565}
6	6	7	{10552, 20538, 29402, 32139, 16260, 21529, 565, 26014}
5	4	7	{10552, 20538, 29402, 32139, 565, 16260, 21529, 26014}
3	0	7	{565, 10552, 16260, 20538, 21529, 26014, 29402, 32139}