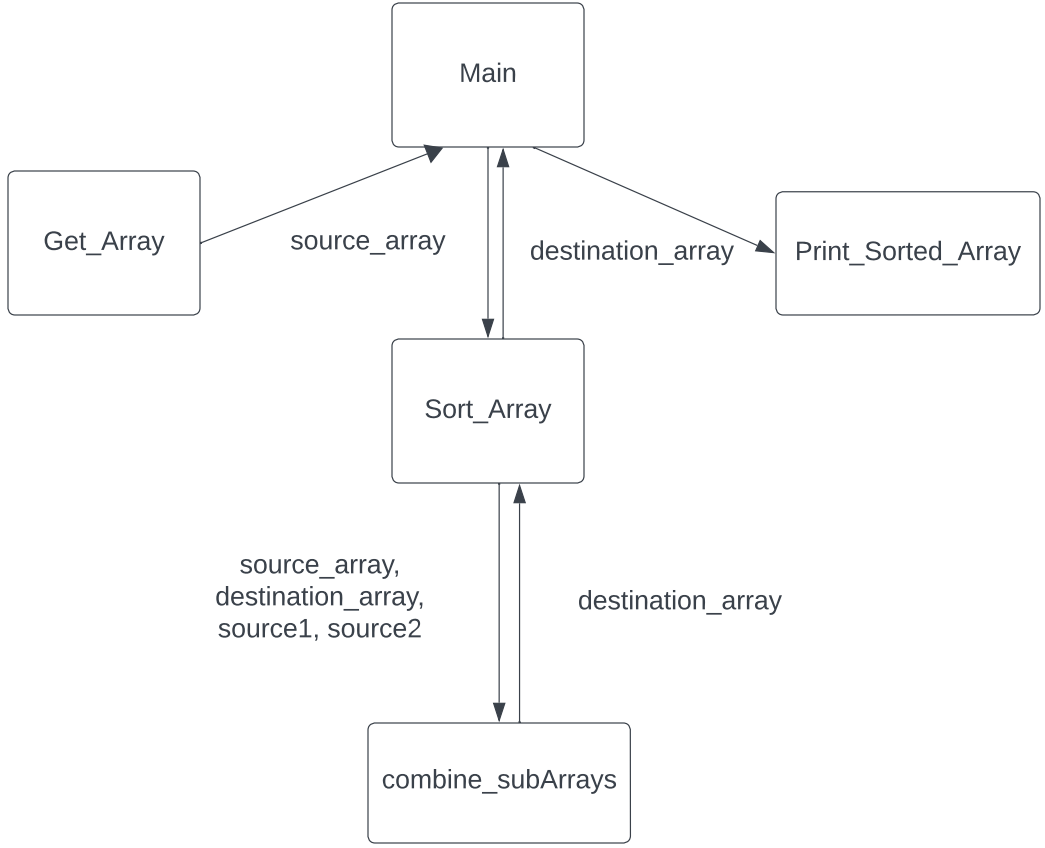
Main
Get_Array
source_array
destination_array
Print_Sorted_Array
Sort_Array
source_array,
destination_array,
source1, source2
destination_array
combine_subArrays

LAB 07\sort.pseudo

```
sort_array(source_array)

    //Creates the source array and an empty array
    array_size => length(source_array)
    destination_array = [0] ...array_size
    number -> 2

    //Sort loop array
    while (number > 2) {
        number -> 0
        source1_begin = 0

        //Create sub arrays
        while (source1_begin < array_size) {
            source1_end = source1_begin + 1
            while (source1_end < array_size and source_array[source1_end - 1] <= source_array[source1_end]) {
                source1_end add 1
            }

            source2_begin = source1_end

            if (source2_begin < array_size) {
                source2_end = source2_begin + 1
            }
            else {
                source2_end = source2_begin + 1
            }


            while (source2_end < array_size and source_array[source2_end - 1] <= source_array[source2_end]) {
                source2_end add 1
            }

            number add 1

            source1 = (source1_begin, source1_end)
            source2 = (source2_begin, source2_end)

            //Calls function to combine source1 and source2
            combine_subArrays(source_array, destination_array, source1, source2)
            source1_begin = source2_end

        }
        //swap array
        source_array, destination_array = destination_array, source_array
    }
    return the source_array


combine_subArrays(source_array, destination_array, source1, source2)

    source1_end = source2_begin
```

```
53
54      // Iterate through the length of arrays
55      for (source1_begin ...source2_end) {
56
57          if (source1_begin < source1_end) and (source2_begin == source2_end or
    source_array[source1_begin] < source_array[source2_begin]) {
58              destination_array[iteration] = source_array[source1_begin]
59              source1_begin add 1
60          }
61
62          else {
63              destination_array[iteration] = source_array[source2_begin]
64              source2_begin add 1
65          }
66      }
67      return the destination_array
68
```