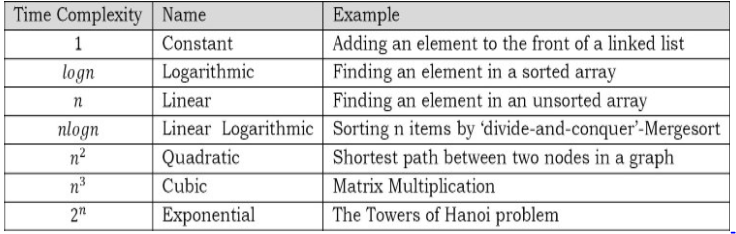
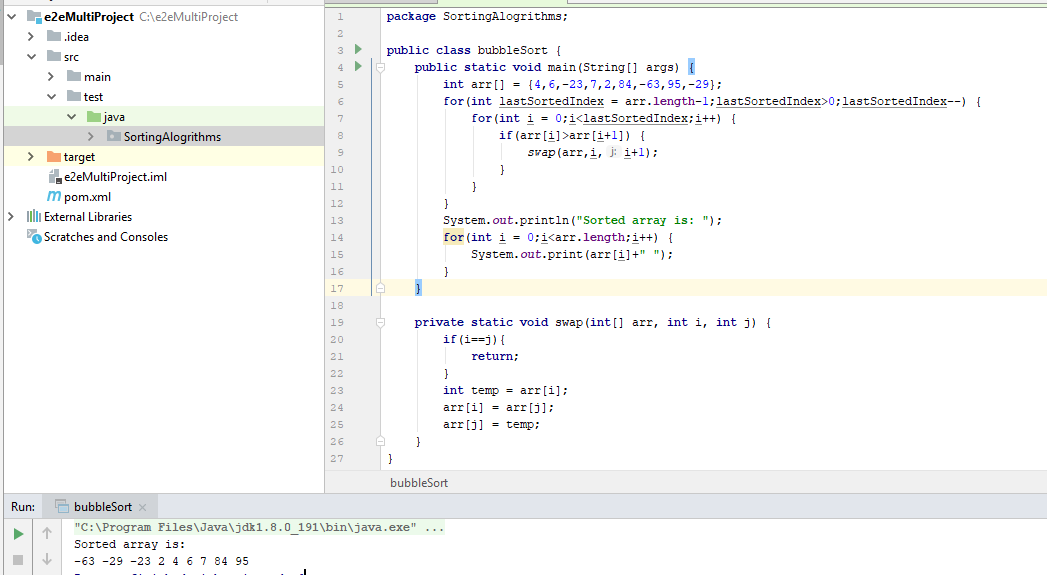
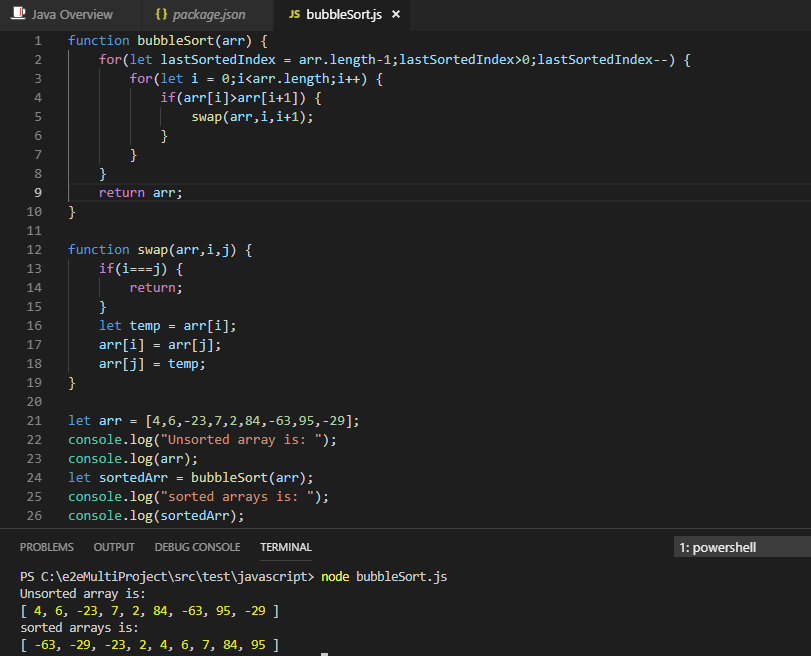
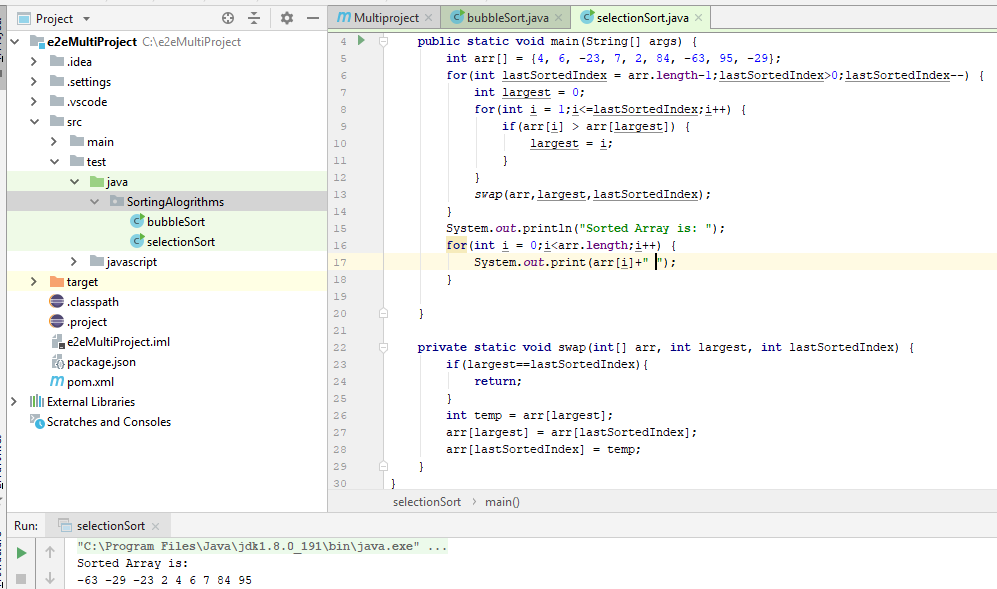
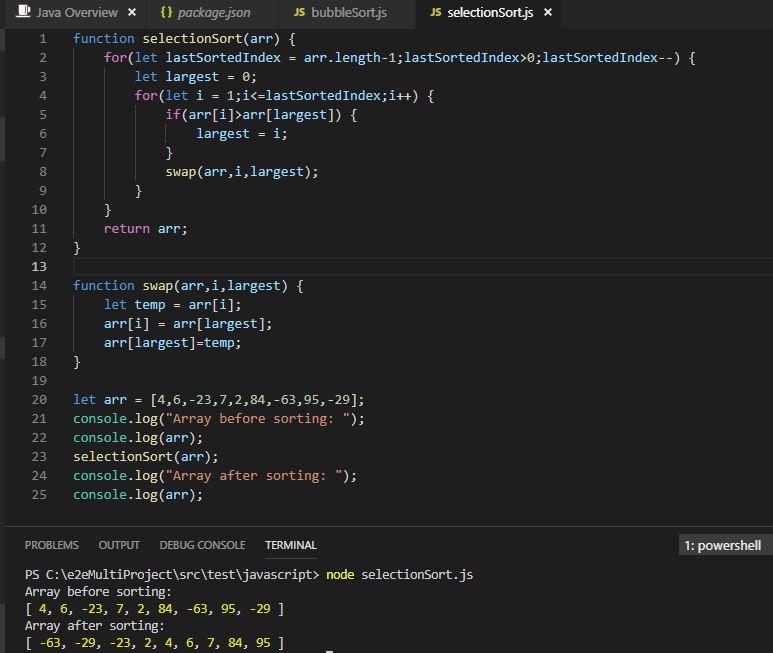
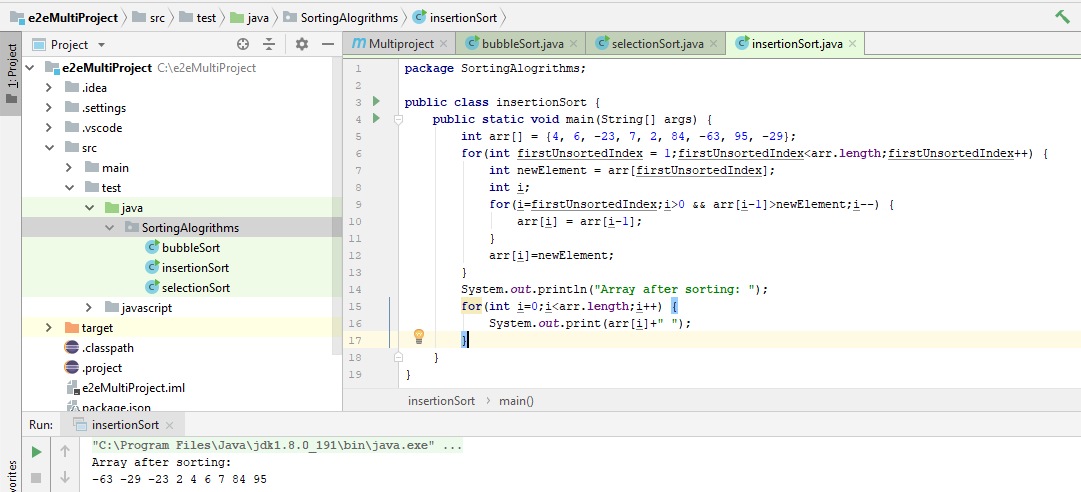
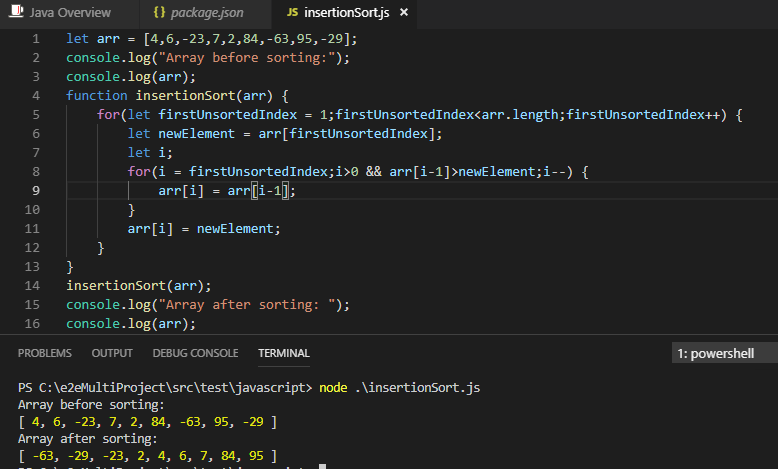
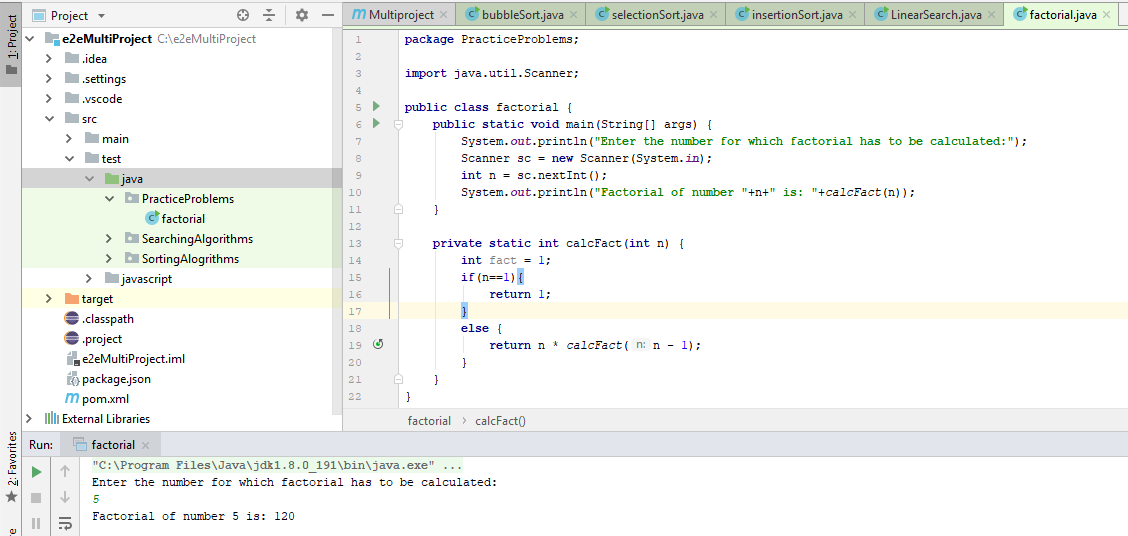
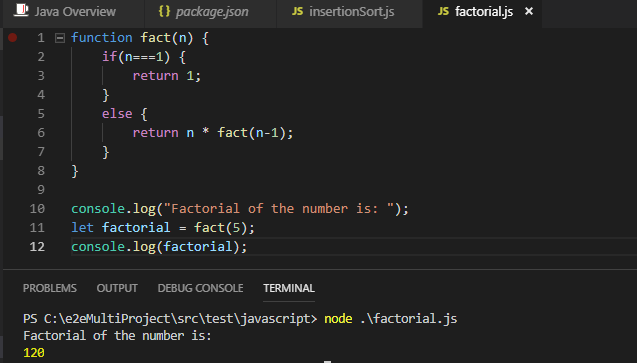
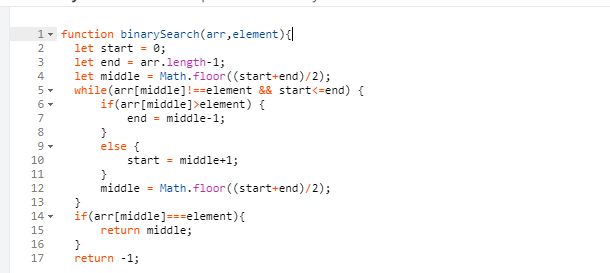
1. 
2. Abstract Data types:  
   Doesn’t mention concrete implementations  
   Stack, Queue, hash map, priority queue
3. Data structures : Organizes and stores data  
   Array, linked list, heap
   1. Big O of Array operations:  
      Insertion -> O(n)  
       O(1) -> to add element at the end  
      deletion -> O(n) -> to delete element by shifting the elements  
       O(1) -> when deleting element at a specific index  
      Searching -> O(1) -> when we know the index  
       O(n) -> when we do not know the index
4. Maps are Abstract Data types
5. Ternary Search Trees: faster than hash maps and support sort function  
   Every node can have 3 children,
6. TST are much more memory efficient, but are not as fast as Tries, so we have to combine for Tries and TST,
7. Binary Search Trees: Search, remove and add operations -> O(N) -> Has root node and leaf nodes
8. Leaf node -> does not have any child nodes
9. Every node has at-most two child nodes, left child is lesser than parent, right child is always greater than the parent node
10. Height of tree is 2^(n-1)
11. Traversal  
    In Order traversal -> Left subtree -> root -> right sub tree  
    Pre-Order traversal -> root -> left sub tree -> right subtree recursively  
    Post Order traversal -> left subtree-> right subtree -> root recursively
12. Bubble Sort Algorithm: performance degrades when the number of elements to sort increases:  
    Find the greatest element and put it to the end of the array in every iteration  
    Java Solution:  
      
      
    Javascript Solution:  
    
13. Stable sort: all items are unique
14. Unstable sort: contains duplicates
15. Selection Sort:

Java Solution:  
  
javascript Solution:  


1. Insertion sort:   
   grows from left to right  
   Useful mostly if array is almost sorted   
   Java Solution:  
     
   JavaScript Solution:
2. 
3. Factorial of a number using recursion:  
   java Solution:  
     
   javascript Solution:  
   
4. Counting Sort:
5. Shell Sort:  
   gap value = (3^k-1)/2
6. Search Algorithms:
7. Binary Search:  
   
8. Quick Sort:  
   Selects a pivot element and places all elements less than pivot to left of pivot, and all the numbers to the right of it greater than the pivot element