1. Ture or False questions:
   a. One can find the maximum subarray of an array with n elements within O(n log n) time;
   b. In the maximum-subarray problem, combining solutions to the sub-problems is more complex than dividing the problem into sub-problems;
   c. Bubble sort is stable;
   d. Merge-sort is in-place;
   e. Divide-and-Conquer algorithms always run faster than iterative algorithms;

Answers:

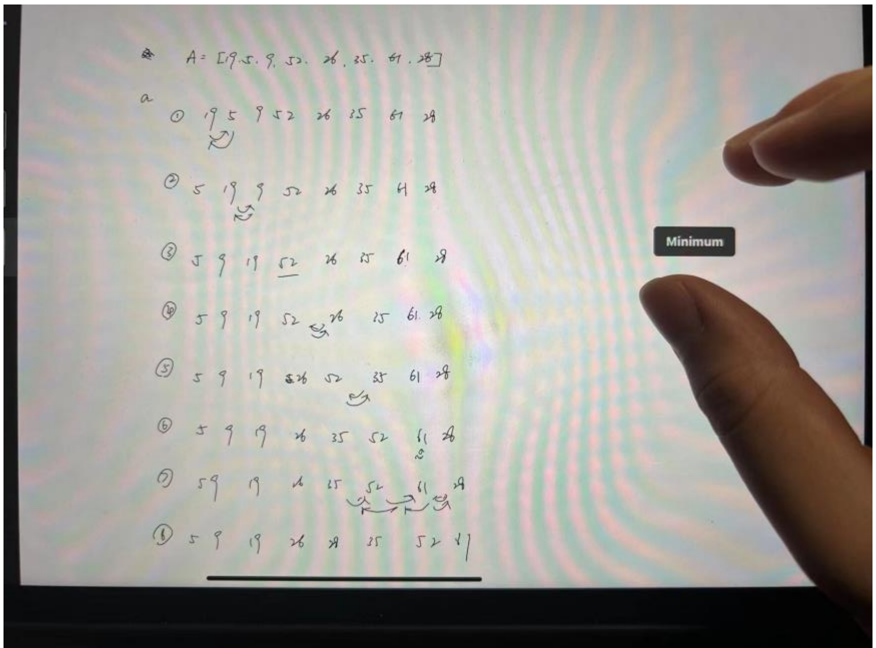
a.F
b.T
c.T
d.F

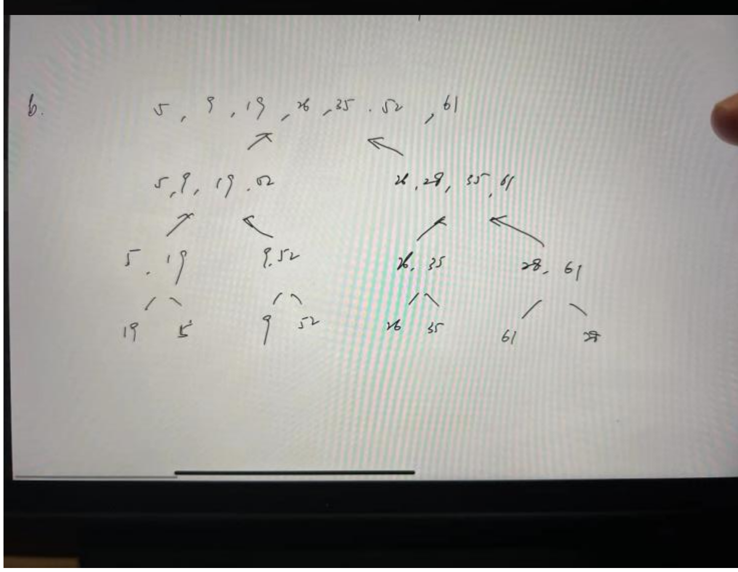
2. Given the array $A[19, 5 , 9, 52, 26, 35, 61, 28]$

   a) Using Figure 2.2 on page 18 of CLRS as a model, illustrate the operation of insertion-sort on A.

   b) Using Figure 2.4 on page 35 of CLRS as a model, illustrate the operation of merge-sort on A.

Answers:





3. Consider sorting n numbers stored in array A by first finding the smallest element of A and exchanging
it with the element in A[1]. Then find the second smallest element of A, and exchange it with A[2].
Continue in this manner for the first n−1 elements of A.
a) Write pseudocode for this algorithm, which is known as selection sort.
b) What loop invariant does this algorithm maintain?
c) Why does it need to run for only the first n−1 elements, rather than for all n elements?
d) Give the best-case and worst-case running times of selection sort in Θ-notation.

a)

```
procedure Selection_sort(A):
    n = length (A)

for j = 1 to n -1:
            smallestVlue = j
            for I = j + 1 to n:
                        if A[i] < A[smallestValue]:
                                    smallestValue = i

            swap A[j] with A[smallestValue]
```

b)

in each outer for loop, the A[1,,,,j-1]consists of the smallest j-1 elements in array[1,…,n]

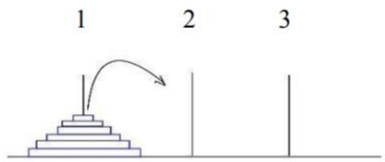
c)

after n-1 elements, the A[1….n-1] have the smallest n-1 elements in the sorted order, therefore, the A[n], should be the biggest value.

d)

the overall running time in the big O notation would be O(n^2) in both best-case and worst -case.

4.



A mathematical game or puzzle consisting of three rods and a number of disks of various diameters, which can slide onto any rod. The puzzle begins with **n** disks stacked on a **start** rod in order of decreasing size, the smallest at the top, thus approximating a conical shape. The objective of the puzzle is to move the entire stack to the **end** rod, obeying the following rules:

1. Only one disk may be moved at a time.
2. Each move consists of taking the top disk from one of the rods and placing it on top of another rod or on an empty rod.
3. No disk may be placed on top of a disk that is smaller than it.

Please design a MOVE(n, start, end) function to illustrate the minimum number of steps of moving n disks from start rod to the end rod.

Q.a: Give the output of MOVE(4, 1, 3).

Q.b: What's the minimum number of moves of MOVE(5, 1, 3), and MOVE(n, 1, 3)?

Answers:

Qa:

```
Move disk 1 from source 1 to destination 2
Move disk 2 from source 1 to destination 3
Move disk 1 from source 2 to destination 3
Move disk 3 from source 1 to destination 2
Move disk 1 from source 3 to destination 1
Move disk 2 from source 3 to destination 2
Move disk 1 from source 1 to destination 2
Move disk 4 from source 1 to destination 3
Move disk 1 from source 2 to destination 3
Move disk 2 from source 2 to destination 1
Move disk 1 from source 3 to destination 1
Move disk 3 from source 2 to destination 3
Move disk 1 from source 1 to destination 2
Move disk 2 from source 1 to destination 3
Move disk 1 from source 2 to destination 3
```

Qb

Minimum of Move(5,1,3) is 31 and the Move(n,1,3) is 2^n-1

Alogirhtm:

```
def move(n, start, end):

     Middle = 6 – start – end

     If n == 1:
          Print(start, end

     Else:
          Move(n-1, start, middle)
          Move(1, start, end)
          Move(n-1, middle, end)
```