DS Tut-10

List 1 = [40, 60, 1, 200, 9, 83, 17]
List 2 = [11, 9, 7, 5, 3, 2]
List 3 = [12, 16, 20, 40, 50, 70]

(1.) Bubble sort

(1) We assume list is an array of n elements. We further assume that swap function swaps the values.

(2)

```
begin  Bubble sort(list)
    For all elements of list
        if [list[i] > list[i+1]
            swap(list[i], list[i+1])
        end of
    end for
return list
```

(3) end Bubble sort

(2) Selection sort

(1) set MIN to location 0
(2) Search the minimum element in list
(3) swap with value location MIN
(4) Increment MIN to point to next element
(5) Repeat until list sorted.

Insertion sort

(1) Iterate from arr[1] to arr[n] over array
(2) Compare current element (key) to its

predecessor.

(3) If the key element s is smaller than its predecessor, compare it to the elements before. Move the greater element one position up to make space for swapped element.

Quicksort

(1) make any element as pivot
(2) Partion the array on the basis of pivot
(3) Apply quick sort on left partition recursively
(4) Apply quick sort on right partion recursively

mergesort.

```
merge_sort (arr, beg, end)

if beg < end
set mid = big + end / 2
merg - sort ( arr, beg, mid)
merge - sort ( arr, mid+1, end )
merge (arr, beg, mid, end)
end of if

End merge sort
```

(2)

Bubble sort

list 1 = [40, 60, 1, 200, 9, 8, 3, 17]

60 40 1 200 9 8 3 17

6 40 1 60 200 9 8 3 17

6 40 1 60 9 200 8 3 17

6 40 1 60 9 200 3 8 17

6 1 40 60 9 3 200 8 17

6 1 40 60 9 3 200 8 17

1 6 40 60 9 3 200 8 17

1 6 40 8 9 60 3 200 8 17

1 6 40 9 3 60 200 8 17

1 6 40 9 3 60 8 200 17

⋮

1 3 8 9 17 40 60 200

list 2 = [11, 9, 7, 5, 3, 2]

B

9 11 7 5 3 2

9 7 11 5 3 2

9 7 5 11 3 2

9 7 5 3 11 2

9 7 5 3 2 11

⋮

2 3 5 7 9 11

selection sort

-> 40 60 1 200 9 8 3 17

1 60 40 200 9 8 3 17
1 3 40 200 9 8 60 12
1 3 8 200 9 40 60 17
1 3 8 9 200 40 60 17
1 3 8 9 17 40 60 200 //.

->

11 9 7 5 3 2

2 9 7 5 3 11
2 3 7 5 9 11
2 3 5 7 9 11 //

Insertion sort.

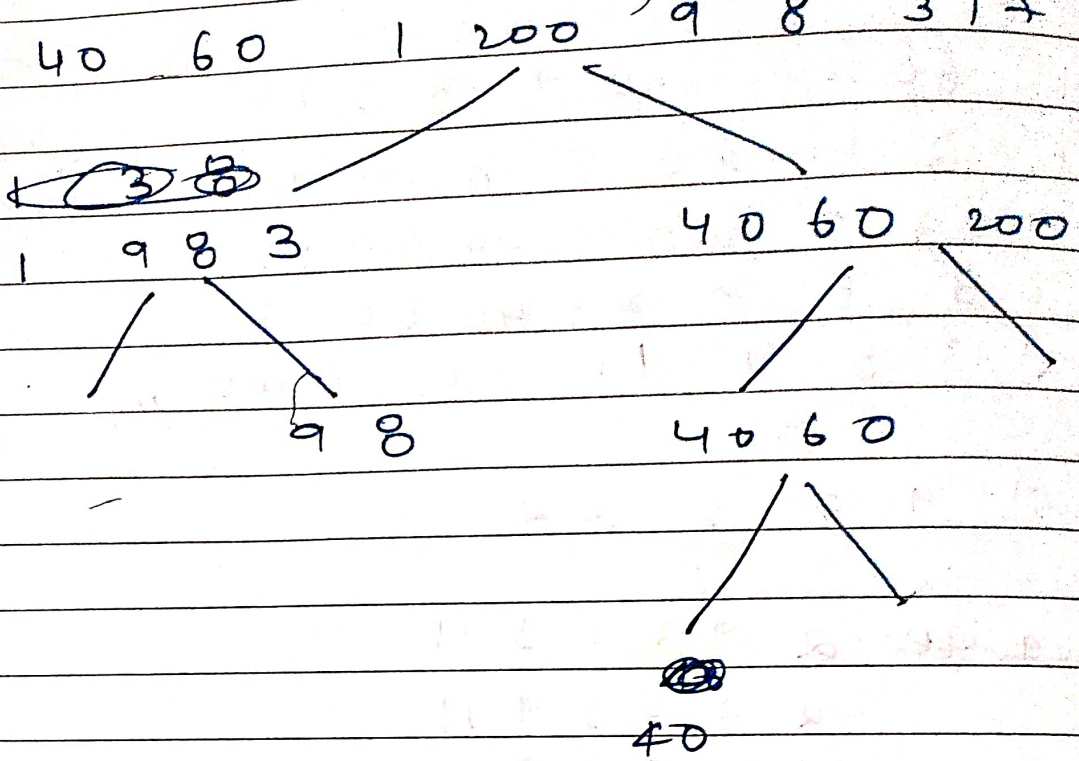
-> 40 60 1 200 9 8 3 17

1 40 60 6 200 9 8 3 17
1 9 40 60 200 8 3 17
1 8 9 40 60 200 3 17
1 3 8 9 40 60 200 17
1 3 8 9 17 40 60 200 //

Quick sort

-> [11 9 7 5 3 2]
9 11 7 5 3 2
9 11 7 5 3 2
7 9 11 5 3 2
5 7 9 11 3 2
3 5 7 9 11 2
2 3 5 7 9 11 //

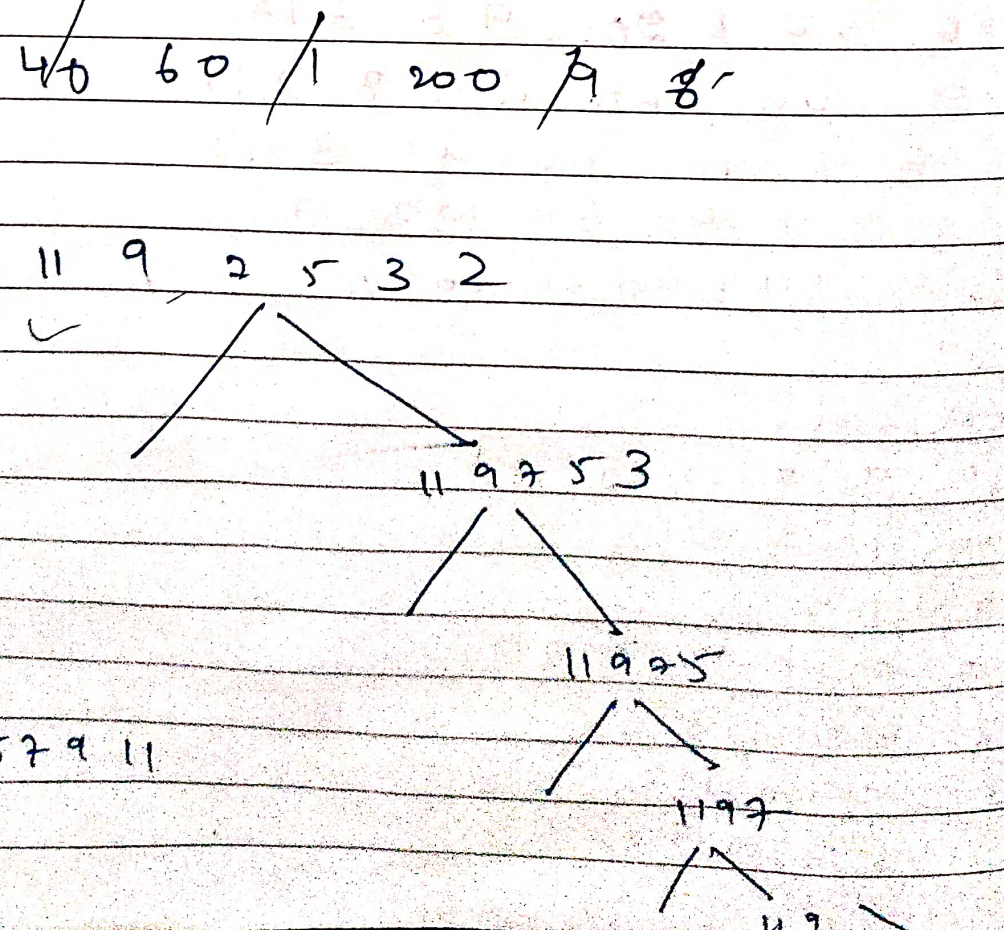
(iv) Quick sort



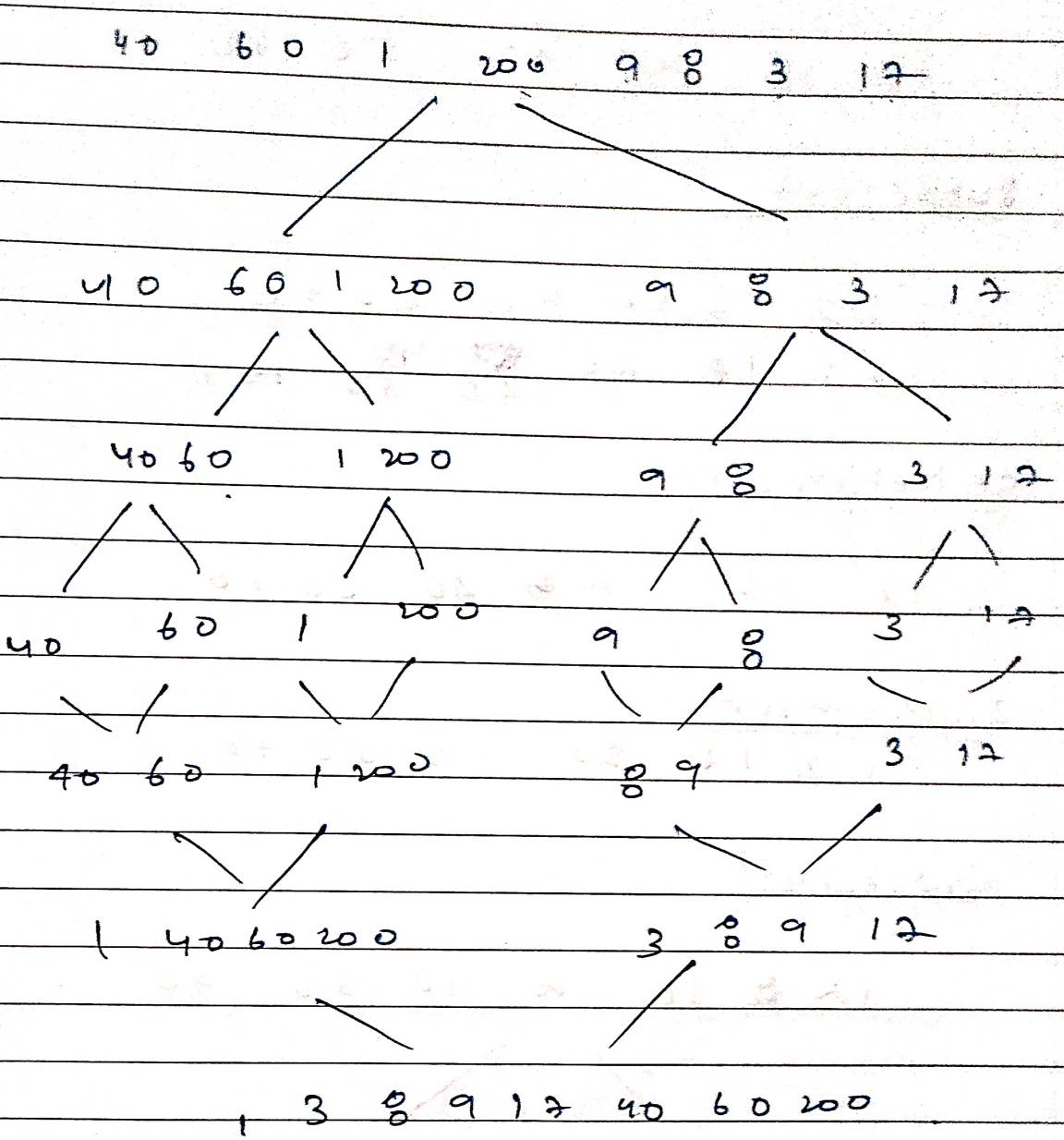
merge.

1 3 8 9 17 40 60 200

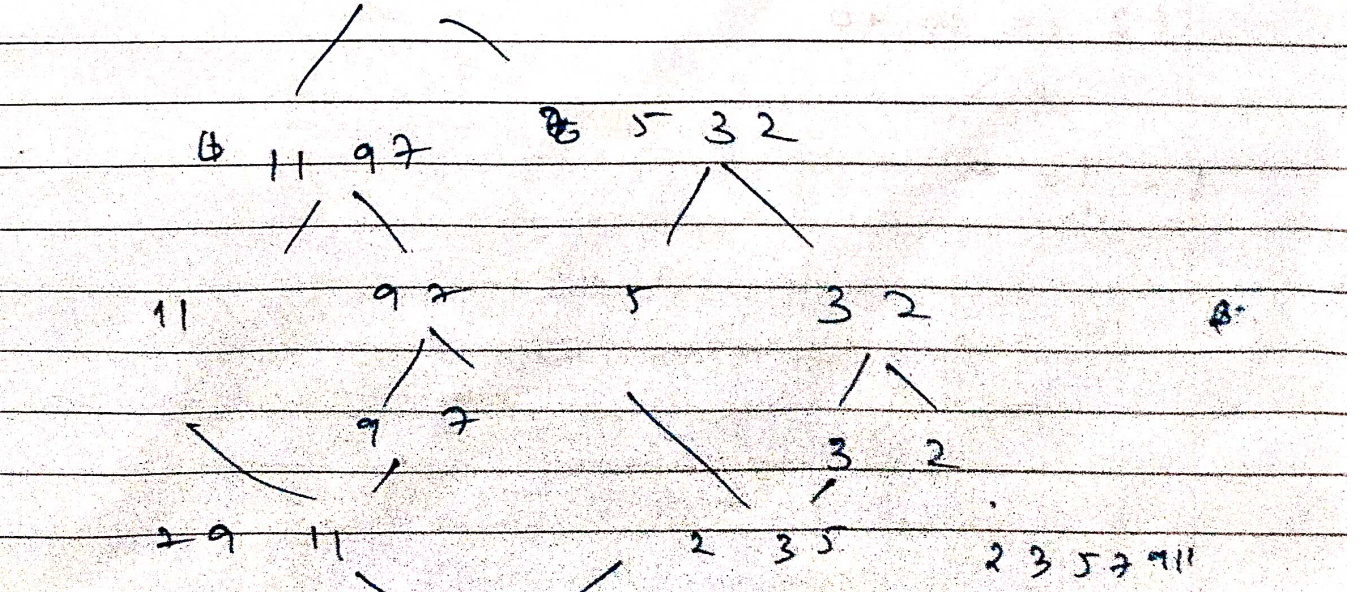
(v) merge sort



(V) merge sort



→ 11 9 7 5 32



(3)

12 16 20 40 50 70

Bubble sort

12 16 20 40 50 70

12 16 20 40 50 70

selection sort

12 16 20 40 50 70

Insertion sort

12 16 20 40 50 70

Quick sort

12 16 20 40 50 70

12 16 20 40 50

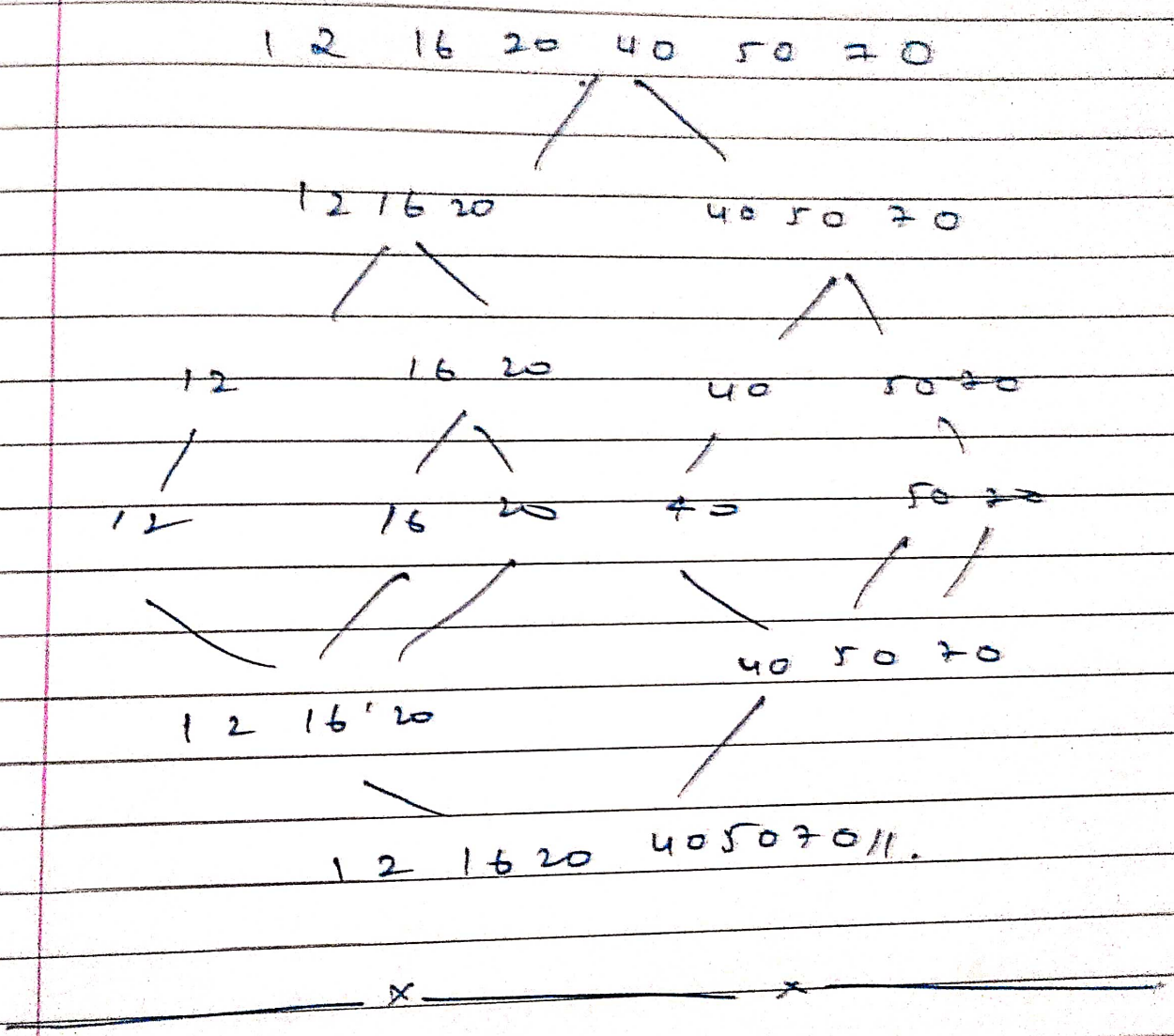
12 16 20 40

12 16

12 16

12

Merge sort


1 2 16 20 40 50 70
1 2 16 20
40 50 70
1 2
16 20
40
50 70
1 2
16 20
40
50
40 50 70
1 2 16 20
1 2 16 20 40 50 70 11.