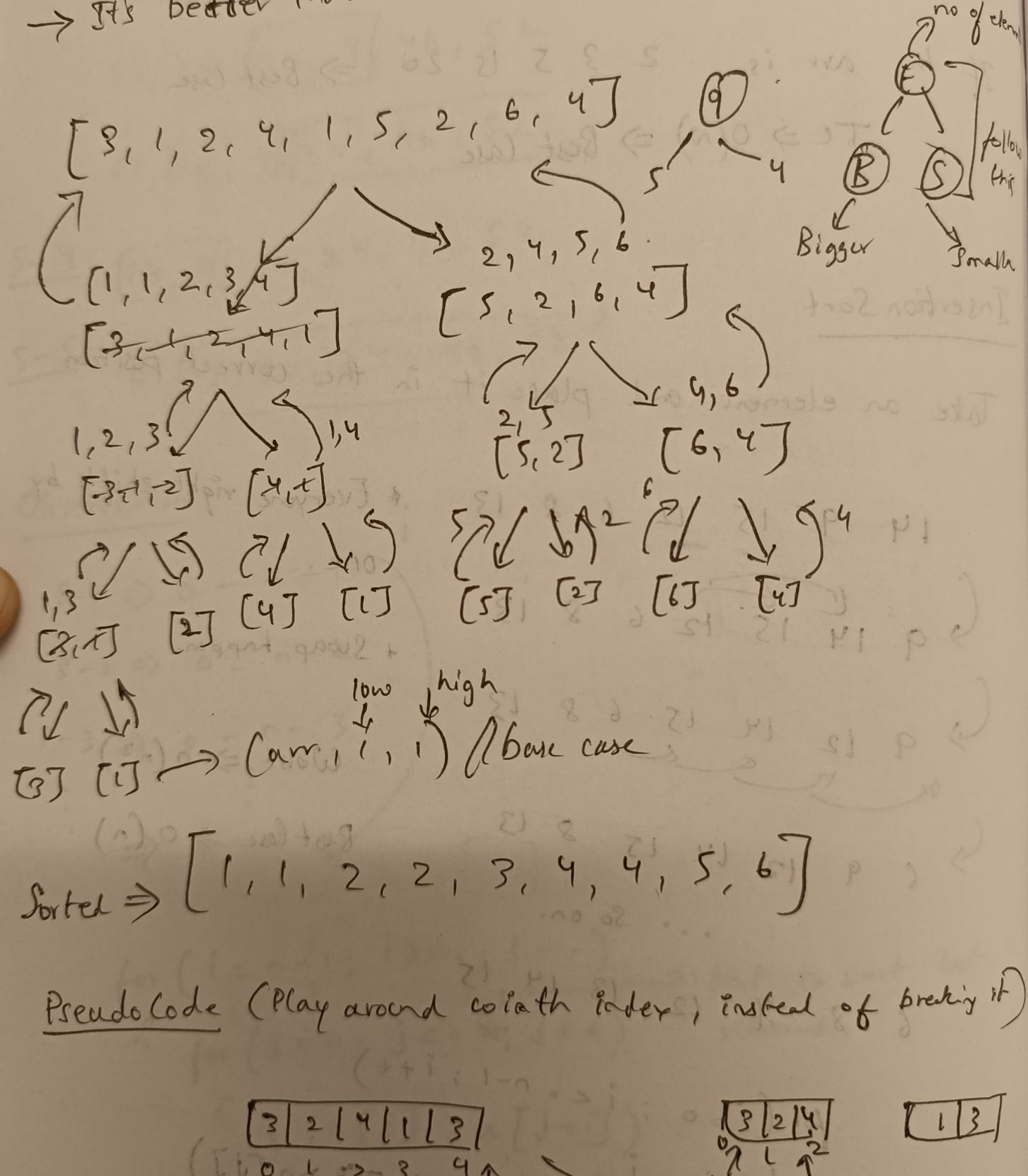
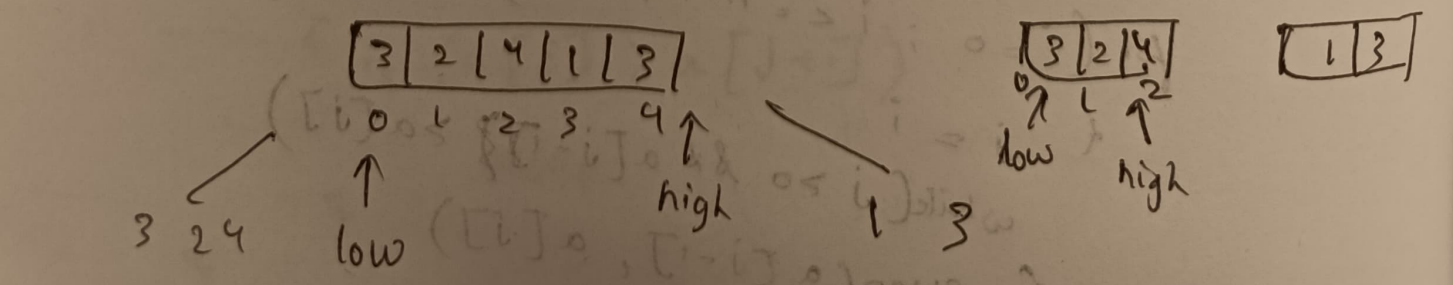
Merge Sort → Divide & Merge.

→ It's better than other sorting algorithm.



Sorted ⇒ [1, 1, 2, 2, 3, 4, 4, 5, 6]

Pseudo Code (Play around coiath index, instead of breaking it)



```
merge_sort(arr, low, high)
{  //base case  if (low >= high) return;
   mid = (low + high)/2
   merge_sort (arr, low, mid)
   merge_sort (arr, mid+1, high)
   merge(arr, low, mid, high);
}
```

Dry Run → 0, 4

```
ms(arr, l, h)
{ if(l >= h) return;
  m = (l+h)/2 → 2
  ms(arr, low, mid) →
  ms(arr, m+1, h)
  m(arr, l, m, h)
}
```

ms(0,2)
{ if( ) x
m = 1
ms

Approach of merge

3, 1, 2, 4, 1, 5, 6, 2, 4
0 1 2 3 4 5 6 7 8

low, mid, mid+1, high

left: 3, 1, 2
left: 1, 2, 4
right: 5, 6, 2, 4
5, 6, 7, 8

```
merge (arr, low, mid, high)
{
    temp -> [ ]
    left = low
    right = mid + 1;
    while (left <= mid && right <= high)
    {
        if (arr [left] <= arr [right])
            temp.add (arr [left])
                left ++;
        else
            temp.add (arr [right])
                right ++;
    }
    while (left <= mid)
    {
        temp.add (arr [left]);
            left ++;
    }
    while (right <= high)
    {
        temp.add (arr [right]);
            right ++
    }
```

```
for( i = low -> high)
{ arr[i] = temp[i - low];
}
```

T.C