Merge Sort Using Recursion

Array using merge sort

recursion call ← 8, 3, 4, 12, 5, 6 → recursion call

1) Divide this array into 2 parts

ends

3, 4, 8 , 5, 6, 12 → two pointers at those two divided array

After these 2 are sorted merge them

3<5, 4<5, 8>5,

[3, 4, 5, 6, 8, 12]

1) Divide array into two parts 2) get both parts sorted via recursion.

3) Merge the sorted parts

arr1 = [3, 5, 9, 19, 32]

arr2 = [4, 6, 8]

* pointer

= [3, 4, 5, 6, 8, 9, 19, 32]

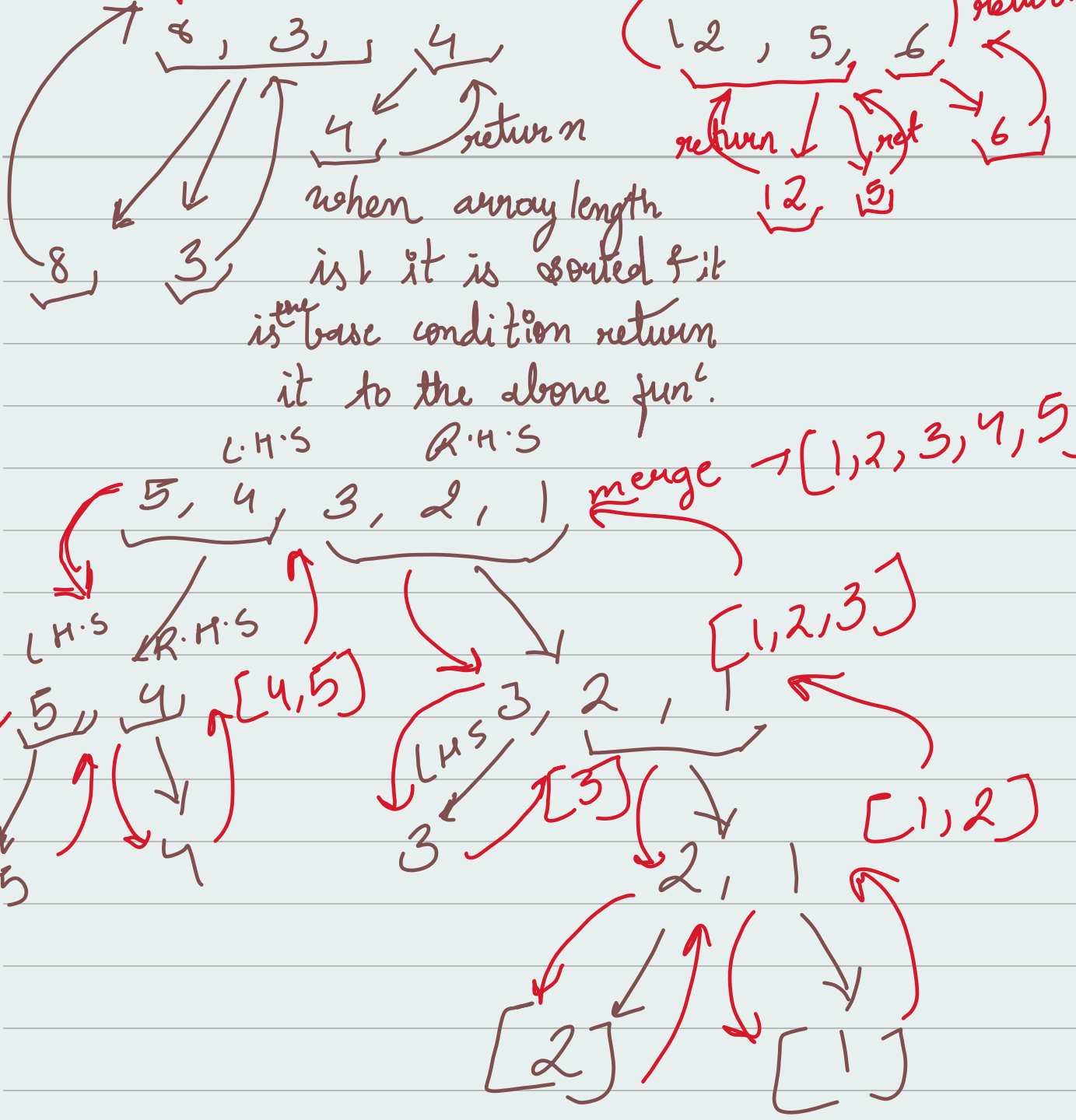
Size (arr1 + arr2)

[3, 4, 8] merge [5, 6, 12] → Return an entire array

8, 3, 4, 12, 5, 6

3, 8 | 4 | 5, 12 | 6

turn

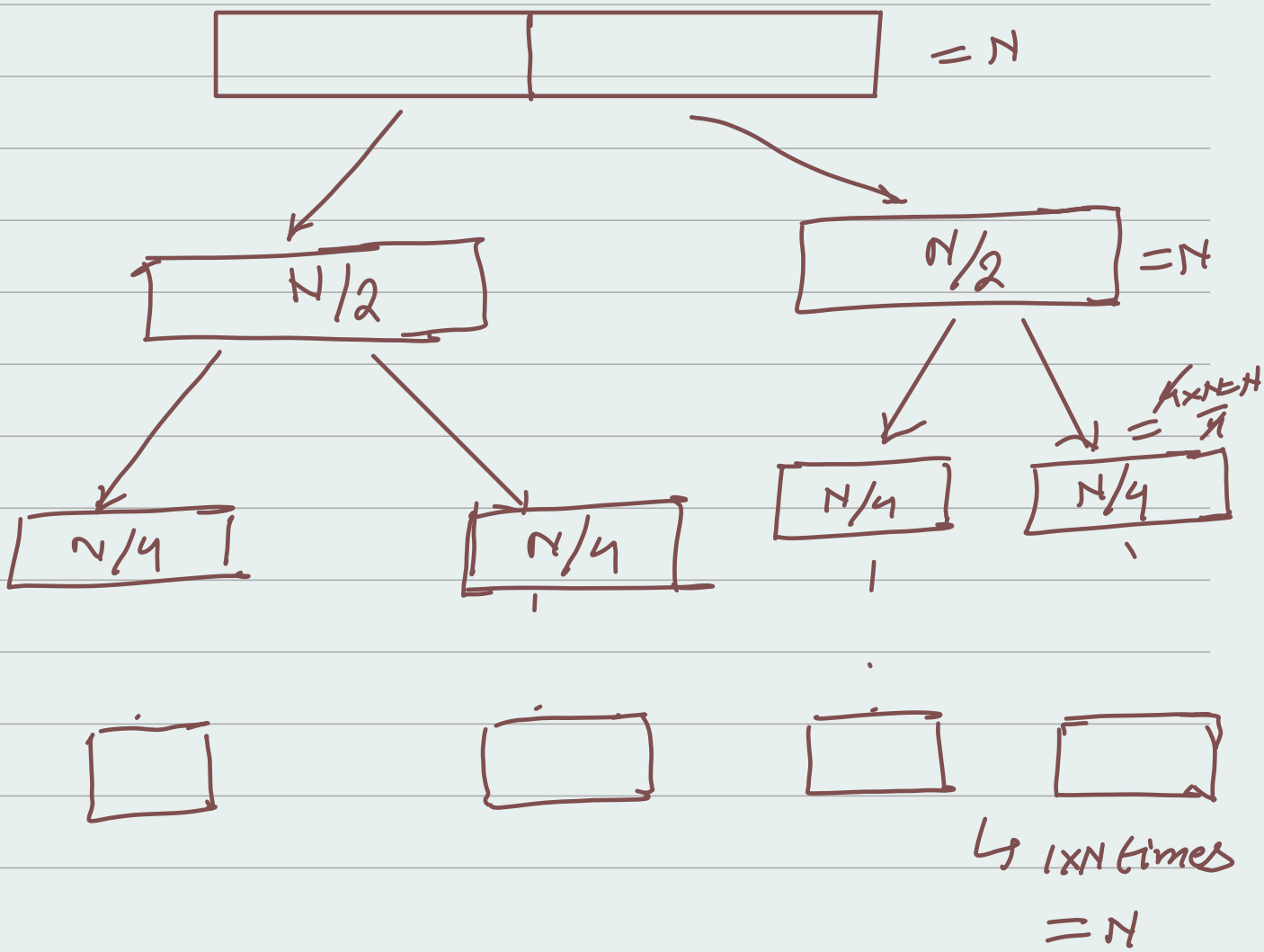



```
psvm ( ) {
int[ ] arr = {8, 3, 4, 12, 5, 6};
  arr = mergeSort (arr);
  System.out.println (Arrays.toString
                      (arr));
}
static int[ ] mergeSort (int[ ] arr){
  // base condition
    if (arr.length == 1){
```

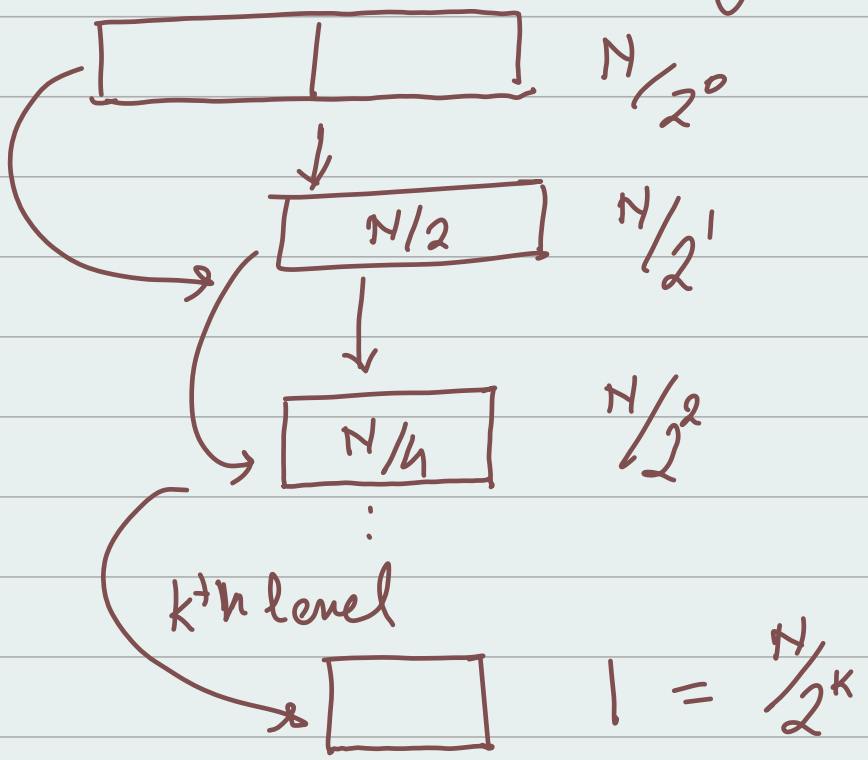
```
    return arr;
}

int mid = arr.length/2;

int[] left = mergeSort(arr, 0, mid);
int[] right = mergeSort(arr, mid, arr.length);

return merge(left, right);
}

private static int[] merge(int[] first, int[] second){
int[] mix = [first.length + second.length];

int i=0;
int j=0;
int k=0;
while (i<first.length && j<second.length){
    if (first[i]<second[j]){
        mix[k] = first[i];
        i++;
        k++;
    }else{
        mix[k] = second[j]
        j++;
        k++;
    }
//It may be possible that one of the array is not
complete then add all the remaining elements
```

in the array, copy all the elements

```
while (i < first length){
    mix[k] = first[i];
    i++;
    k++;
}
while (j < second length){
    mix[k] = second[j];
    j++;
    k++;
}
    return mix;
}
```



* At every level (N) elements are being merged



$$1 = \frac{N}{2^k}$$

$$2^k = N$$

$$k \log 2 = \log N$$

$$\boxed{k = \log_2 N}$$

Total complexity = Total level × work at every level

$$= N \times \log(N)$$

Time complexity $\Rightarrow O(N \log(N))$

Space complexity → Auxiliary SP $\Rightarrow O(N)$

These many comparisons are required to merge two arrays

$$T(N) = T(N/2) + T(N/2) + (N-1)$$

$$= 2T(N/2) + (N-1)$$

$$2 \times \frac{1}{2^P} = 1 \qquad \text{Hence } P \Rightarrow 1$$

$$T(N) = x + x \int_1^x \frac{u-1}{u^2}\, du$$

$$\int_1^x \left(\frac{1}{u} - \frac{1}{u^2}\right) du$$

$$\int_1^x \frac{du}{u} - \int_x^1 \frac{du}{u^2}$$

$$\log u - \int u^{-2}\, du$$

$$\log u - \frac{u^{-1}}{-1} \Rightarrow \log u + u^{-1}$$

$$\Rightarrow \left[\log u + \frac{1}{u}\right]_1^x$$

$$= \log x + \frac{1}{x} - 1$$

$$T(N) = x + x\left[\log x + \frac{1}{x} - 1\right]$$

$$T(N) = x + x \log x + 1 - x$$

$$T(N) = x \log x + 1$$

$$O(x \log(x))$$

$$T(N) = O(\log(N))$$

In-place Sorting of MergeSort

s=0, 1, m=2, 3, e=4

5, 4, 3, 2, 1

(arr, 0, 4)

(arr, start, mid) (arr, mid, end)

i: 4, 5 (s) j: 1, 2, 3 (mid, e) (arr, s, m, e)

for (k=0; k < length(mix); k++){ mix=[1,2,3,4,5]

arr[s+k] = mix[k]

Ex. index: 0 1 2 3 4

4 5 3 2 1

arr[s+k] = arr[3+0] = mix[0]

arr[s+k] = arr[3+1] = 2 3

1 2 , 1 2

s+0 s+1

[2, 1]

[2] [1]