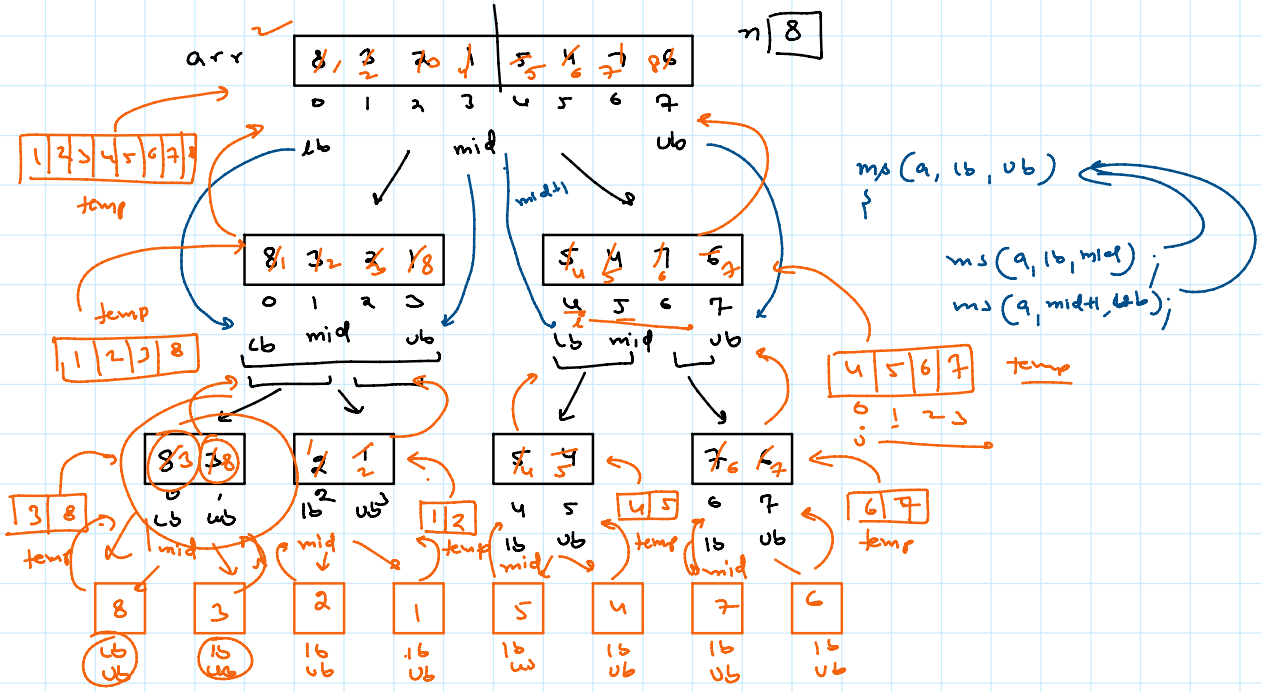
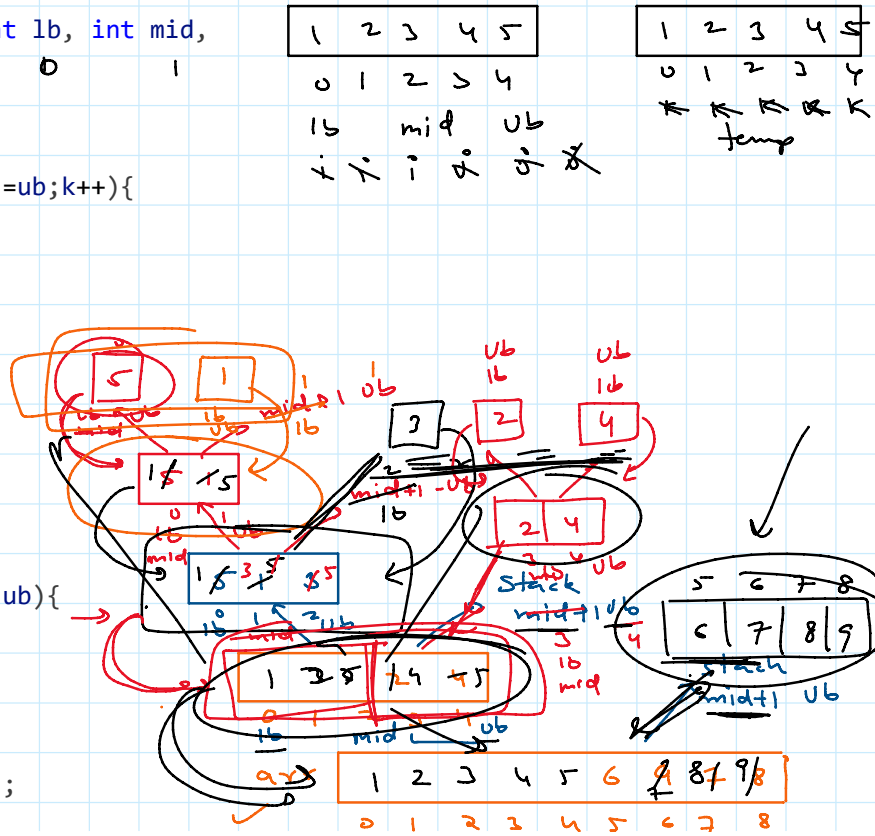
DAY 14

09 May 2024 01:06 PM

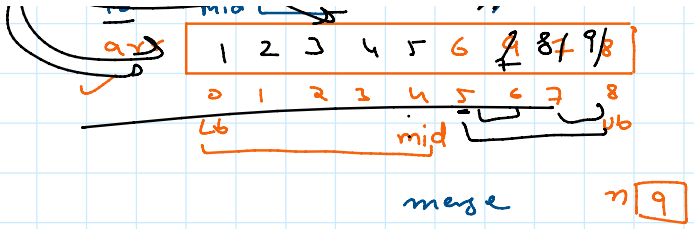
Merge sorting Using Recursion :-



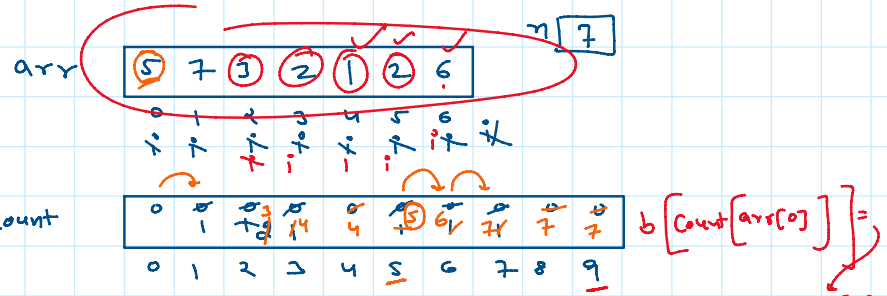

```
#include<iostream>
using namespace std;
void mergeTwoSortedParts(int arr[], int lb, int mid,
int ub)
{
    int temp[ub-lb+1];
    int i,j,k;
    for(i=lb,j=mid+1,k=0;i<=mid and j<=ub;k++){
        if(arr[i]<arr[j])
            temp[k]=arr[i++];
        else
            temp[k]=arr[j++];
    }
    while(i<=mid)
        temp[k++]=arr[i++];
    while(j<=ub)
        temp[k++]=arr[j++];
    for(i=lb,j=0;i<=ub;i++,j++)
        arr[i]=temp[j];
}
void mergeSort(int arr[], int lb, int ub){
    if(ub<=lb)
      return;
    int mid=(lb+ub)/2;
    mergeSort(arr,lb,mid);
    mergeSort(arr,mid+1,ub);
    mergeTwoSortedParts(arr,lb,mid,ub);
}
```

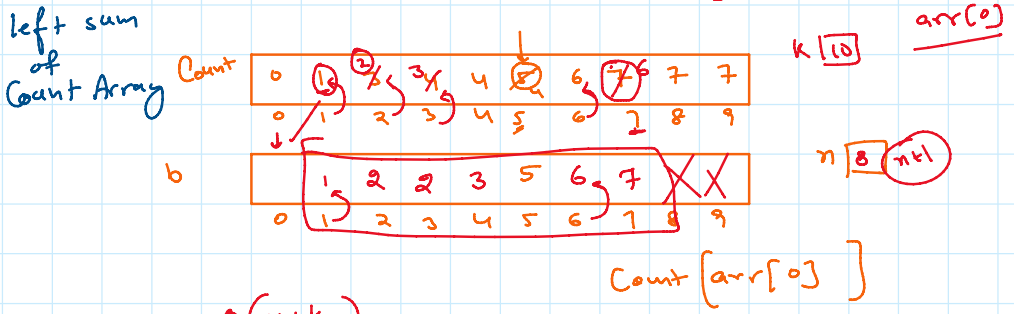


```
    mergeSort(arr,mid+1,ub);
    mergeTwoSortedParts(arr,lb,mid,ub);
}
int main()
{
    int arr[]={5,1,3,2,4,6,9,7,8};
    int n=sizeof(arr)/sizeof(int);
    mergeSort(arr,0,n-1);
    for(int i:arr)
    {
        cout<<i<<" ";
    }
    return 0;
}
```



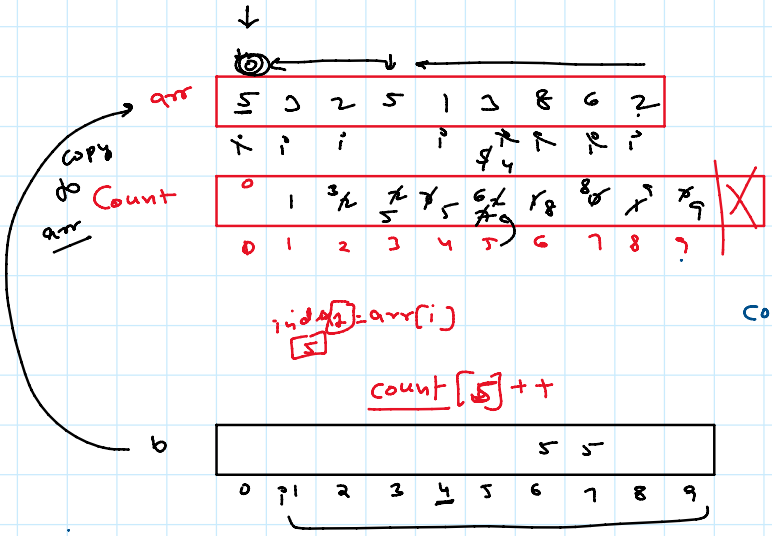
Counting sort :-





$O(n+k)$

↑ length of count Array

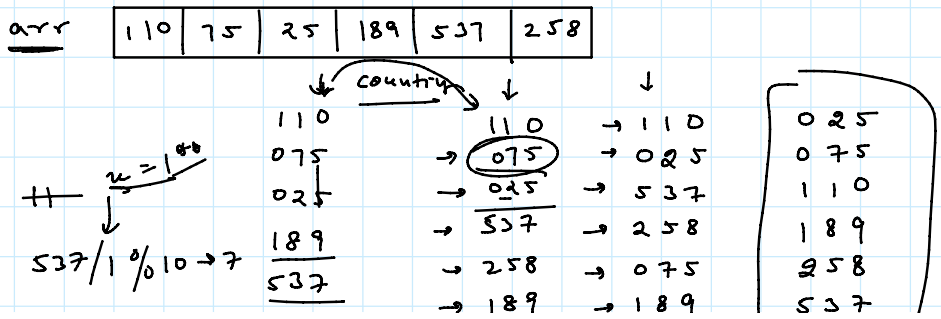



```
b[count[arr[i]]]=arr[i];
```

count[arr[i]]--;

```cpp
#include<iostream>
using namespace std;
void countingSort(int arr[], int n)
{
    int count[10]={0};
    int b[n+1];
    //count elements
    for(int i=0;i<n;i++)
    {
        int index=arr[i];
        count[index]++;
    }
    //left sum of count array
    for(int i=1;i<10;i++)
    {
        count[i]+= count[i-1];
    }
    //new index
    for(int i=0;i<n;i++)
    {
        b[count[arr[i]]]=arr[i];
        count[arr[i]]--;
    }
    //copy from b to arr
    for(int i=0;i<n;i++)
    {
        arr[i]=b[i+1];
    }
}
int main()
{
    int arr[]={5,1,3,2,4,6,9,7,8};
    int n=sizeof(arr)/sizeof(int);
    countingSort(arr,n);
    for(int i:arr)
    {
        cout<<i<<" ";
    }
    return 0;
}
```

Radix sort :→



537/1 % 10 → 7

537/10 % 10 → 3

537/100 % 10 → 5

1000

537 → 258 → 189

189 → 258 → 537

189
258
537

arr[i]/1000

075/1 % 10 → 5

075/10 % 10 → 7

075/100 % 10 → 0

15

14 10

13 9 6

12 8 5 3

11 7 4 2 1