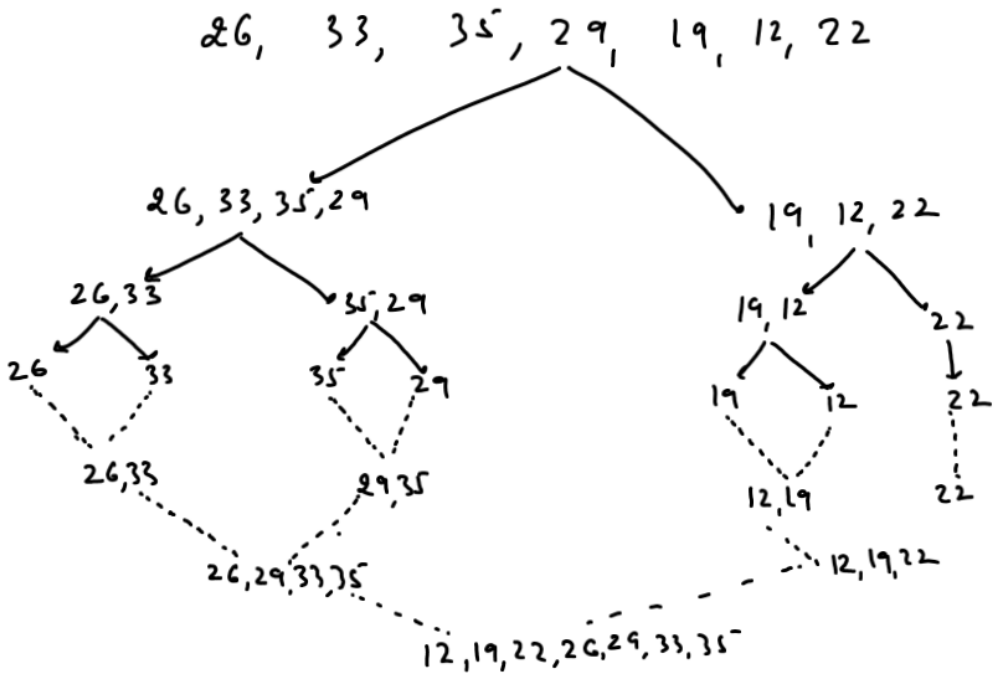
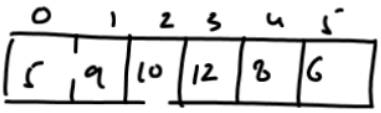
MERGE SORT (Divide And Conquer Approach)



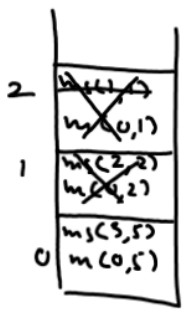



```
void merge_sort(int arr[],int first,int last)
{
    int mid;
    if(first<last)
      {
         mid=(first+last)/2;
       ✓ merge_sort(arr,first,mid);
        merge_sort(arr,mid+1,last);
        merge(arr,first,last);
      }
}
```

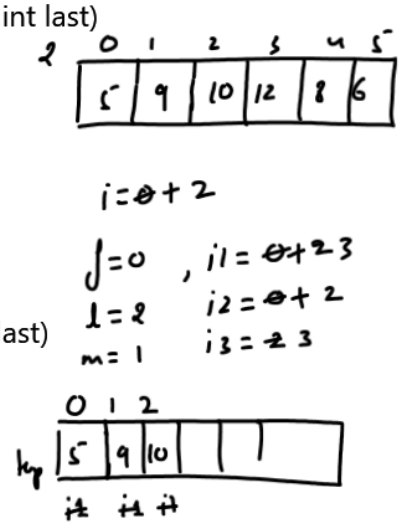
a. f=0, l=5
m = 2
ms(arr, 0, 2)

b. f=0, l=2
m = 1
ms(arr, 0, 1)

c. f=0, l=1
m = 0
ms(arr, 0, 0)

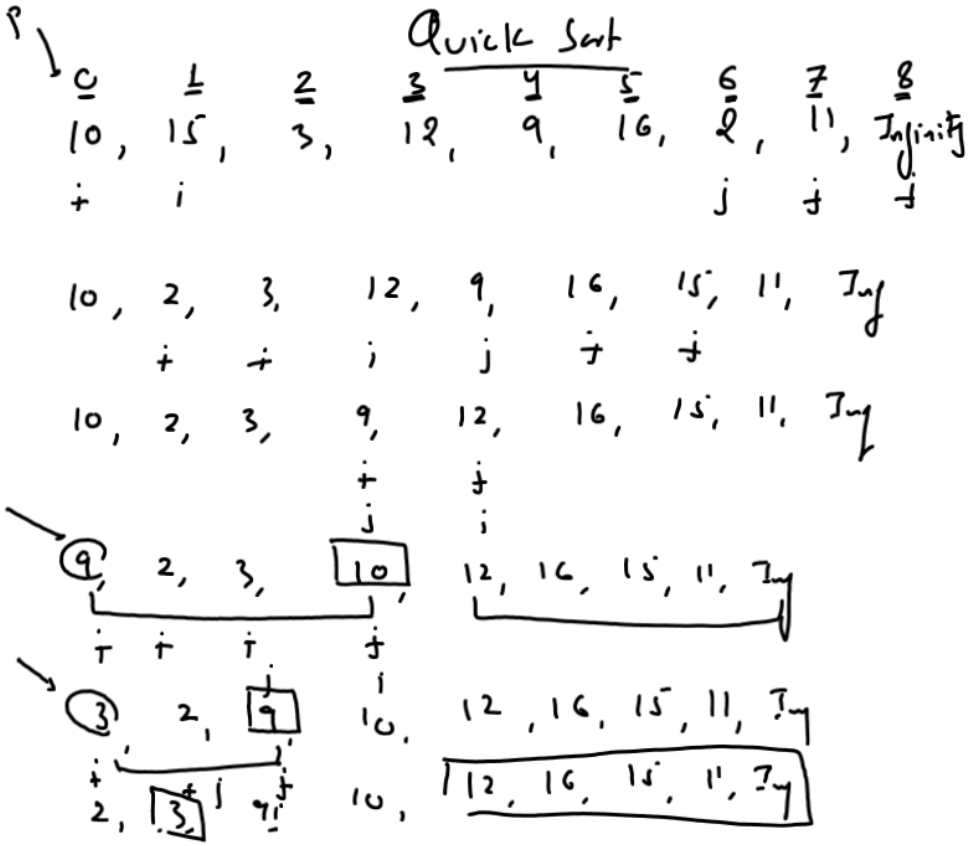


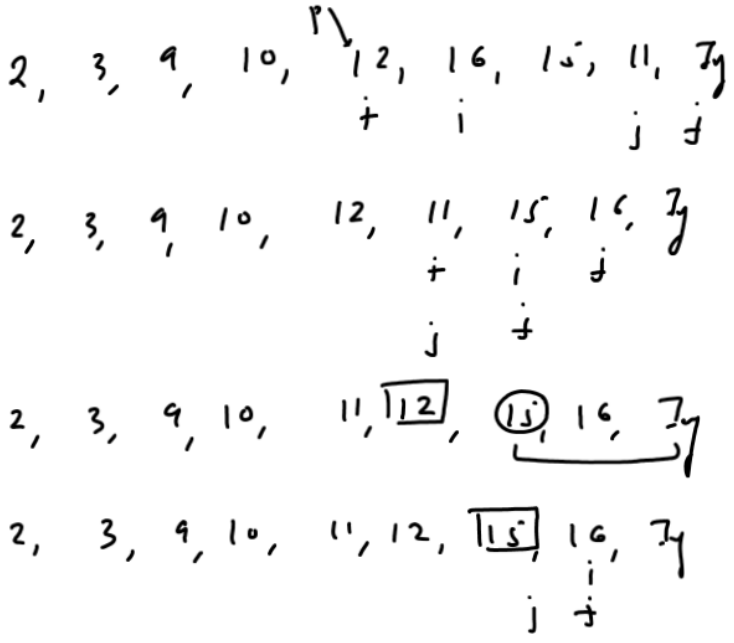
```
void merge(int arr[],int first,int last)
{
   int temp[10];
   int mid,i,i1,i2,i3;
   mid=(first+last)/2;
   i1=0;
   i2=first;
   i3=mid+1;
   while(i2<=mid && i3<=last)
   {
     if(arr[i2]<arr[i3])
      {
        temp[i1]=arr[i2];
        i2=i2+1;
      }
     else
      {
        temp[i1]=arr[i3];
        i3=i3+1;
      }
  i1=i1+1;
  }
if(i2<=mid)
 {
   while(i2<=mid)
   {
     temp[i1]=arr[i2];
     i1=i1+1;
     i2=i2+1;
   }
}
else
{
  while(i3<=last)
   {
     temp[i1]=arr[i3];
     i1=i1+1;
     i3=i3+1;
   }
 }
 i=0;
 while(i<i1)
 {
   arr[first+i]=temp[i];
   i=i+1;
  }
}
```



```
int main()
{
    int arr[]={10,5,9,12,8,6};
    printf("Before sorting:");
    for(int i=0;i<6;i++)
    {
        printf("\n%d",arr[i]);
    }
    merge_sort(arr,0,5);
    printf("\nAfter sorting:");
    for(int i=0;i<6;i++)
    {
        printf("\n%d",arr[i]);
    }
   return 0;
}
```

Quick Sort


p
0 1 2 3 4 5 6 7 8
10, 15, 3, 12, 9, 16, 2, 11, Infinity
i j
10, 2, 3, 12, 9, 16, 15, 11, Inf
i j
10, 2, 3, 9, 12, 16, 15, 11, Inf
j i
9, 2, 3, 10, 12, 16, 15, 11, Inf
3, 2, 9, 10, 12, 16, 15, 11, Inf
2, 3, 9, 10, 12, 16, 15, 11, Inf


p
2, 3, 9, 10, 12, 16, 15, 11, Inf
i j
2, 3, 9, 10, 12, 11, 15, 16, Inf
i j
j i
2, 3, 9, 10, 11, 12, 15, 16, Inf
2, 3, 9, 10, 11, 12, 15, 16, Inf
i
j

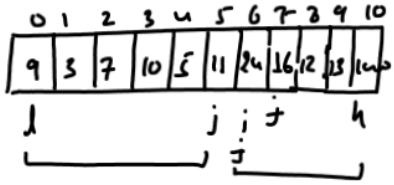
```
int main()
{
 int A[]=
{11,13,7,12,16,9,24,5,10,3,1000}
,n=10,i;

 QuickSort(A,0,n);

 for(i=0;i<10;i++)
 printf("%d ",A[i]);
 printf("\n");

 return 0;
}
void QuickSort(int A[],int l,int h)
{
 int j;

 if(l<h)
 {
 j=partition(A,l,h);
 QuickSort(A,l,j);
 QuickSort(A,j+1,h);
 }
}
```



pivot = 11

```
int partition(int A[],int l,int h)
{
 int pivot=A[l];
 int i=l,j=h;

 do
 {
       do
       {
          i++;
       }while(A[i]<=pivot);
       do
       {
          j--;
       }while(A[j]>pivot);

       if(i<j)
          swap(&A[i],&A[j]);

}while(i<j);

 swap(&A[l],&A[j]);
 return j;
}
```

10, 20, 30, 40, 50, 70

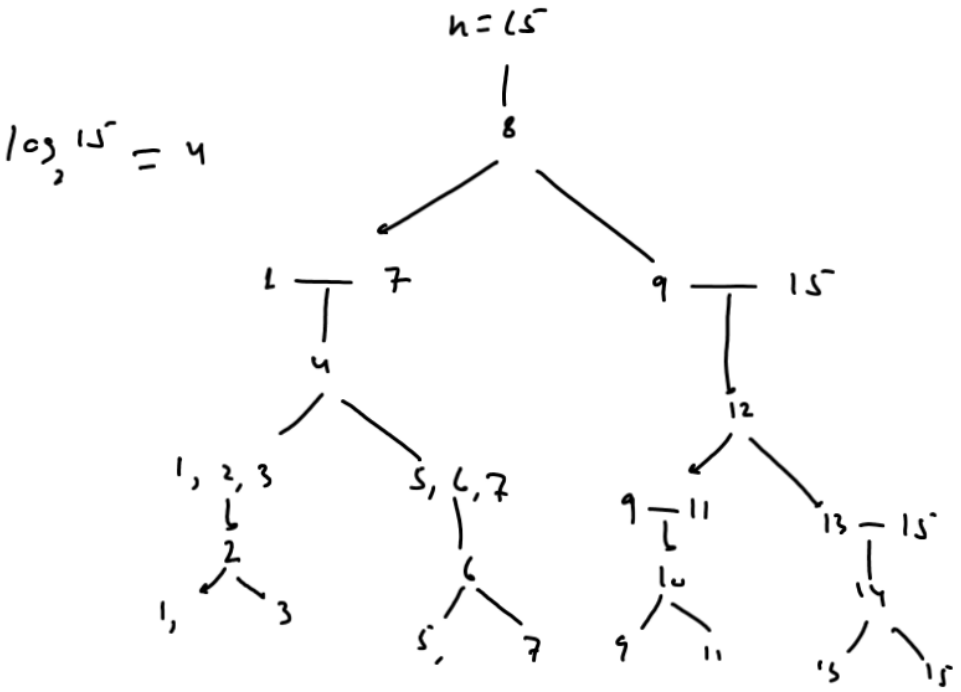
i j

$$C(n) = n + (n-1) + (n-2) + \cdots + 2 + 1$$

$$C(n) = \frac{n(n+1)}{2}$$

$$C(n) = O(n^2)$$

$\log_2 15 = 4$



$$n, \frac{n}{2}, \frac{n}{4}, \frac{n}{8} \cdots \frac{n}{2^k}$$

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

$$\boxed{k = \log n}$$