A	9	3	7	5	6	2
	l					h

Merge Sort

```
Algorithm MergeSort(l, h)
  if (l < h)
      mid = (l+h)/2                 1
      Mergesort(l, mid)             $T(\frac{n}{2})$
      Mergesort(mid+1, h)           $T(\frac{n}{2})$
      Merge(l, mid, h)              n
```

$$T(n) = \begin{cases} 1 & n=1 \\ 2T(\frac{n}{2})+n & n>1 \end{cases}$$

therefore $O(n \log n)$

```
merge(x[1...k], y[1...l])
    if  k=0, return y[1...l]
    if  l=0, return x[1...k]
    if  x[1] ≤ y[1] then
        return x[1]∘merge[x[2...k], y[1...k])
  else
        return y[1]∘merge(x[1...k], y[2...l])
```

Runtime is $O(n)$

Quick Sort

(students ordering each other by height)

//smallest goes back, then tallest goes front, other's arrange each other

```
Partition(l, h)
   pivot = A[l]
   i=l ; j=h;
  do i++ while A[i] ≤ pivot
  do j-- while A[j] > pivot
 if (i<j) swap(A[i], A[j])
  swap(A[l], A[j]);
  return j
```

```
Quicksort(l, h)
if l<h   j = partition(l, h)
     Quicksort(l, j)
     Quicksort(j+1, h)
end if
```