

Selection sort

select min in array[0...] and put in array[0]
 select min in subarray [1...] & put in array[1]
 ...
 process $n-1$
 worst case=best $n*n = n-1 + n-2 + \dots + 2 + 1$
 case=average

Bubble sort: swapping adjacent element, instead select and swap once. slower than selection sort in average, but best case is better (n instead $n*n$ for select

Insertion sort

a[0] is treated as sorted part arr[1...] is treated as unsorted part
 each unsorted is inserted into sorted part in order
 processes $n-1$
 worst case (reversed ordered) $n*n$
 best case (sorted in order) n

Merge sort

1 split into 2 part 2 recursive sort left and right part
 3 leaf node has 1 or 2 elements 4 and merge
 disadvantage temporary arrays, extra space
 advantage fast
 process $n \log(n)$
 cost $n * \log(n)$
 worst case=best input not affect
 case=average performance

Quick Sort

partition recursive sort
 array by pivot value
 scan from both end, swap the bigger on the left to the smaller on the right, until left and right reach the same index, then swap a[pivotposition] with a[0]
 best case fastest sort
 Worst case split into 0, 1..n-1 always, sorted array using a[0] as pivot
 become recursive selection sort
 shuffle or select middle of first several element as pivot
 worst case is very inefficient

Compare Sort algorithm

for small n, select and insert sort used, $n \sim 7$, machine dependent
 for larger n, divide and conquer sort used, until reach a small number.
 in Java, sort array with object type requires the object class must have compareTo() overridden
 Sorting evaluation: CPU time, memory used, array size (Merge sort(larger) --> quick sort(small) --> <7 select/insert sort)
 sorting process and intermediate results -- on test
 comparison, swap or change, space requirement

Sequential search

best case 1
 worst case n
 average $n/2$
 be careful with the code: index

Binary Search

Array must be sorted in searching key
 if n is not power of two, worst case $\log(n)$ with n round to power(z,n)
 $>a[n-2], <a[1]$
 if n is power(2), $\log(n) + 1$
 worst case
 $>a[n-2]$
 $<a[1]$ is $\log(n)$, 1 less
 fully understand the binary sort passes and cost.
 The final is either equals an element a[middle] or not in the range.
 split subarray does not include a[mid]