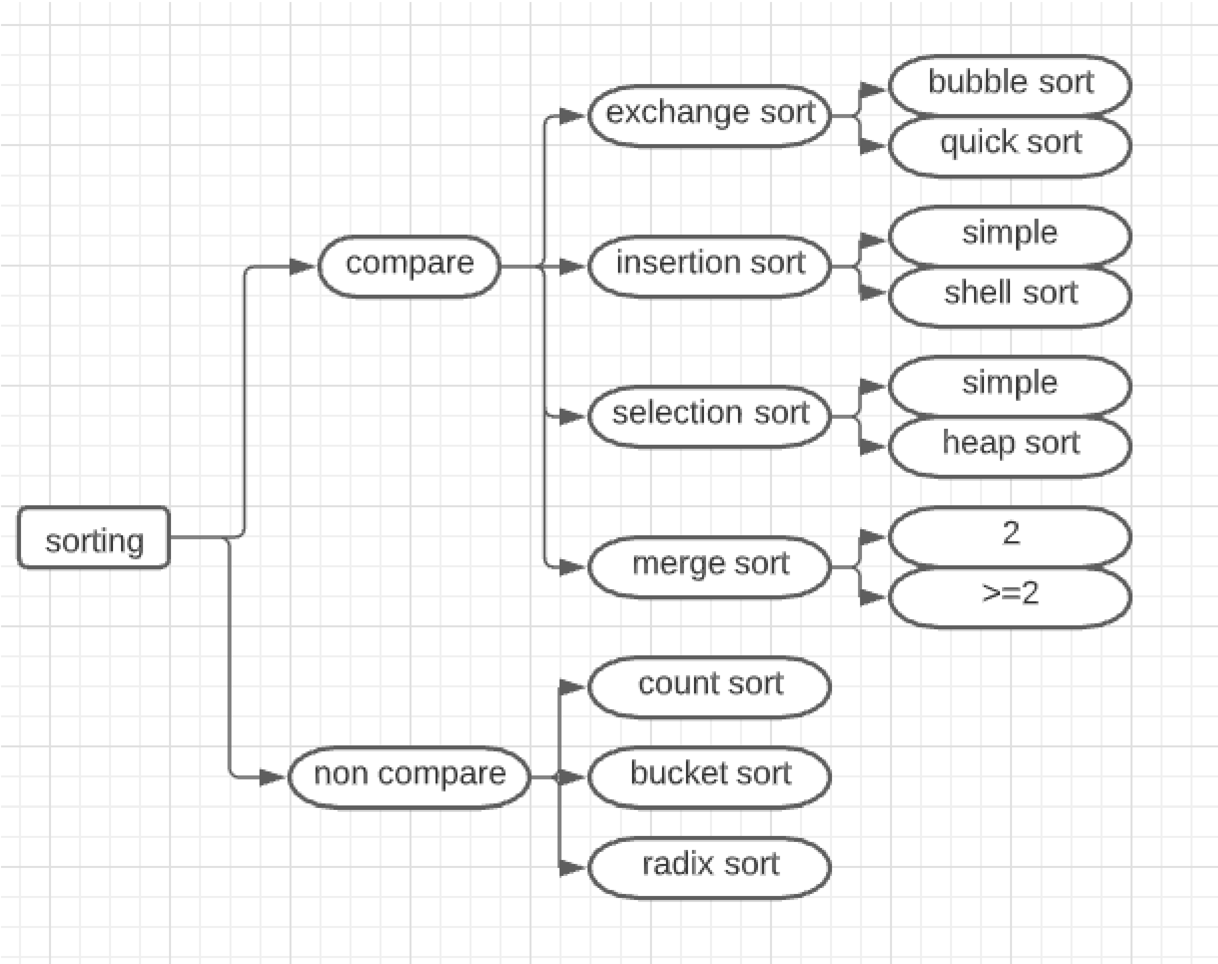
Algorithms - 3 Sorting

Summary of sorting algorithm

why sorting?
Sorting Problem
https://www.cnblogs.com/onepixel/p/7674659.html



Name	Average Time	Worst Time	Best Time	Space	Stability
Insertion Sort	$O(n^2)$	$O(n^2)$	O(n)	O(1)	Yes

Name	Average Time	Worst Time	Best Time	Space	Stability
Merge Sort	$O(nlog_2n)$	$O(nlog_2n)$	$O(nlog_2n)$	O(n)	Yes
Heap Sort					

Details of each sorting

1. Insertion Sort: O(n^2)

```
INSERTION-SORT(A):
    for j in range(2, len(A)): # after every loop A[:j] is sorted
        key = A[j]
        i = j - 1
        while i> 0 and A[i] > key:# find a suitable place for key
            A[i+1] = A[i]
            i = i-1
            A[i+1] = key
```

2. Merge Sort: O(nlogn)

```
MERGE-SORT(A,p,r):
    if p < r:  # check base case
        q = floor((p+r)/2)
        MERGE-SORT(A,p,q)
        MERGE-SORT(A,q+1,r)
        MERGE(A,p,q,r)

MERGE(A,p,q,r):
    n1 = q-p+1
    n2 = r-q
    let L[1...n1+1] and R[1..n2+1] be new arrays   # copy two original list
    for i = 1 to n1:
        L[i] = A[p+i-1]
    for j = 1 to n2:
        R[j]- A[q+j]
    L[n1+1] = infinite
    R[n2+1] = infinite
    for k = p to r: # put back into A in order
        if L[i] <= R[j]:
            A[k] = L[i]
            i = i+1
        else A[k] = R[j]
            j = j+1
```

3. Quick Sort

```
QUICKSORT(A, p, r)
    if p < r
        q = PARTITION(A, p, r)
        QUICKSORT(A, p, q-1)
        QUICKSORT(A, q+1, r)

PARTITION(A, p, r)
    x = A[r]
```

```
    i = p - 1
    for j = p to r - 1
        if A[j] <= x
            i = i + 1
            exchange A[i] with A[j]
    exchange A[i+1] with A[4]
    return i + 1
```

worst $O(n^2)$
To avoid the bad case in QuickSort: Randomization

Averagce case analysis:
$ln(n+1) < \sum_{i=1}^{i=n} \frac{1}{n} < ln(n) + 1$
Let A be $z_1, z_2, \dots, z_n$ in sorted order
$Z_{ij} = z_i, z_{i+1}, \dots, z_j$
Observe that any two elements will be compared at most one

$X_{ij} = \begin{cases} 0 & else \\ 1 & if\ z_i\ is\ compared\ to\ z_i \end{cases}$
Total number of comparisons $X = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} X_{ij}$
$E[X] = E[\sum_{i=1}^{n-1} \sum_{j=i+1}^{n} X_{ij}] = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} E[X_{ij}] = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} Pr(z_i\ is\ compared\ to\ z_j)$
$Pr(z_i\ is\ compared\ to\ z_j) = Pr(z_i\ or\ z_j\ is\ the\ first\ chosen\ pivot\ from\ Z_{ij} compared\ to\ z_j)$
$= Pr(z_i\ is\ the\ first\ chosen\ pivot\ from\ Z_{ij} compared\ to\ z_j) + Pr(z_j\ is\ the\ first\ chosen\ pivot\ from\ Z_{ij} compared\ to\ z_j)$
$= \frac{1}{j-i+1} + \frac{1}{j-i+1} = \frac{2}{j-i+1}$
so, $E[x] = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \frac{2}{i-i+1}$
rename $k = j - i$, then $E[x] = \sum_{i=1}^{n-1} \sum_{k=1}^{n-i} \frac{2}{k+1} < \sum_{i=1}^{n-1} \sum_{k=1}^{n} \frac{2}{k} = \sum_{i=1}^{n-1} [2 \sum_{k=1}^{n} \frac{1}{k}](harmonic\ series) = \sum_{i=1}^{n-1} O(logn)$

Extend:
different partition https://cs.stackexchange.com/questions/11458/quicksort-partitioning-hoare-vs-lomuto