[Data Structure] Summary10

**Lecture 9: Sorting**

**Quick sort**

**텍스트이(가) 표시된 사진

자동 생성된 설명**

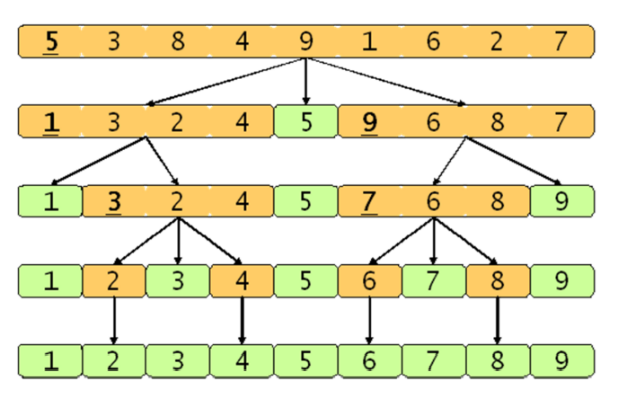
- process

1. If the size of the segment is greater than 1

2. Partition into two lists based on pivot. The ‘partition’ function returns the position of the pivot. 3. Recursive call from left to right before the pivot (except pivot)

4. Recursive call from left next the pivot to right (except pivot)

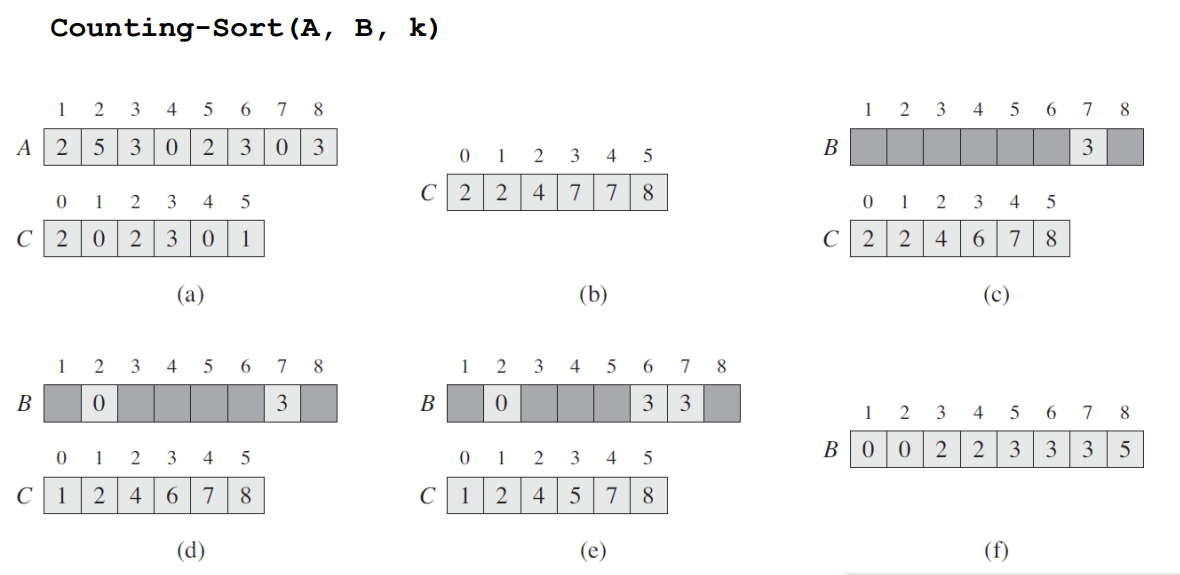
- timecomplexity

**Best case : O(lgn)**

**Worst case : O(n2)**

**Counting Sort**

* stable ; Numbers with the same value appear in the output array in the same order as in the input array

****

**: counting sort** is an [algorithm](https://en.wikipedia.org/wiki/Algorithm) for [sorting](https://en.wikipedia.org/wiki/Sorting_algorithm) a collection of objects according to keys that are small positive [integers](https://en.wikipedia.org/wiki/Integer); that is, it is an [integer sorting](https://en.wikipedia.org/wiki/Integer_sorting) algorithm. It operates by counting the number of objects that possess distinct key values, and applying prefix sum on those counts to determine the positions of each key value in the output sequence. Its running time is linear in the number of items and the difference between the maximum key value and the minimum key value, so it is only suitable for direct use in situations where the variation in keys is not significantly greater than the number of items.

-timecomplexity : O(n+k). usually, k=O(n)

Then, How can we sort them when k is large? -> Radix sort!!

**Radix Sort**

**테이블이(가) 표시된 사진

자동 생성된 설명**

sort the number for each digit

– Sorting order does matter!

– Most significant digit (MSD) vs. Least significant digit (LSD)

– Sort the least significant digit (LSD) first

**: radix sort** is a non-[comparative](https://en.wikipedia.org/wiki/Comparison_sort) [sorting algorithm](https://en.wikipedia.org/wiki/Sorting_algorithm). It avoids comparison by creating and [distributing](https://en.wikipedia.org/wiki/Distribution_sort) elements into buckets according to their [radix](https://en.wikipedia.org/wiki/Radix). For elements with more than one [significant digit](https://en.wikipedia.org/wiki/Significant_digit), this bucketing process is repeated for each digit, while preserving the ordering of the prior step, until all digits have been considered.

-timecomplexity : O(d(n+k))

**Summay of sort algorithm**

**테이블이(가) 표시된 사진

자동 생성된 설명**

**Lecture 10: Graph**

**Graph**

: A data structure that represents the relationships between connected objects. Most common data structure type. The tree is a special case of the graph(no loop)

텍스트, 장치, 봉투, 벡터그래픽이(가) 표시된 사진

자동 생성된 설명

* Vertex : vertices of graph G.Also called a node.
* Edge : Connection between vertices. Also called a link

**Graph Type**

**-**Undirected graph : Can go both ways through edges

-Directed graph : Can go only one way through edges

**Weighted Graph**

**:** A graph with the cost or weight assigned to edges

-Weight: length of road (or time to pass)

**Sub-graph**

**:** A graph consisting of a subset of the vertex set V(G) and the edge set E(G)

**Degree of Graph**

Adjacent vertex : Vertices directly connected by edges at one vertex

1.Degree of undirected graph : The number of edges at each node

2. Degree of directed graph

– In-degree: the number of incoming edges

– Out-degree: the number of outgoing edges

**Path of Graph**

Path: a set of vertices that connect two vertices.

**Graph ADT**

Operation:

– Create\_graph() :: = Create a graph.

– init(g) :: = Initialize graph g. – insert\_vertex(g, v) :: = insert vertex v into graph g.

– insert\_edge(g, u, v) :: = Insert the edge (u, v) into the graph g.

– delete\_vertex(g, v) :: = Delete the vertex v of the graph g.

– delete\_edge(g, u, v) :: = Delete the edge (u, v) of the graph g.

– is\_empty(g) :: = Make sure the graph g is empty.

– adjacent(v) :: = returns a list of vertices adjacent to vertex v.

– destroy\_graph(g) :: = Remove graph g.

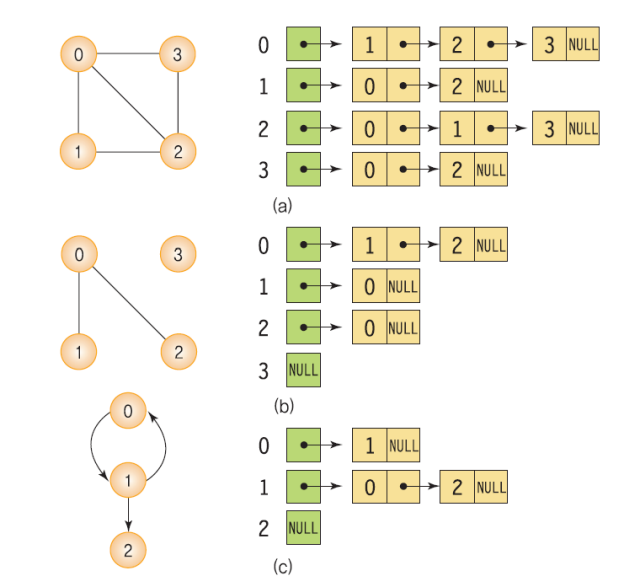
**Graph Representation**

1. Adjacency matrix : if edge(i, j) exists in the graph, M[i][j] = 1,

**텍스트이(가) 표시된 사진

자동 생성된 설명**

**2.** Adjacency list : Represent vertices adjacent to each vertex as a linked list

****