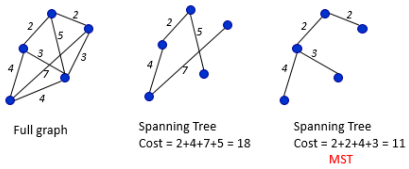
Quizlet

Algorithms

Study online at quizlet.com/_3hpox

g

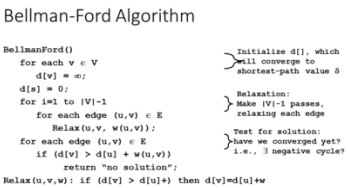
1. **Algorithms for Finding Strongly Connected Components**



STRONGLY CONNECTED COMPONENT ALGORITHM:

In the mathematical theory of directed graphs, a graph is said to be strongly connected or diconnected if every vertex is reachable from every other vertex. The strongly connected components or diconnected components of an arbitrary directed graph form a partition into subgraphs that are themselves strongly connected. It is possible to test the strong connectivity of a graph, or to find its strongly connected components, in linear time.

2. **Bellman-Ford**



SHORTEST PATH ALGORITHM:

The Bellman-Ford algorithm is an algorithm that computes shortest paths from a single source vertex to all of the other vertices in a weighted digraph.

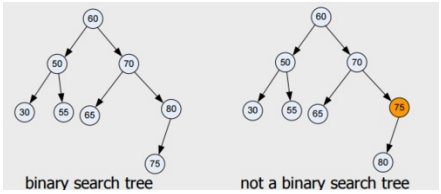
Bellman-Ford is useful for pathfinding when edges may have negative costs. For example if you're navigating a maze with potions which boost health and hazards which lower it, Bellman-Ford would be a great approach.

3. **Big-O Notation**

Complexity	General form	Examples
Constant	$O(n^0)$ or $O(1)$	Is number odd / even
Logarithmic	O(log n)	Binary search, half each iteration
Linear	O(n)	Linear search
Polynomial	$O(n^a)$	$O(n^2)$, $O(n^3)$ etc. Bubble sort
Exponential	$O(a^n)$	$O(2^n)$. Recursive Fibonacci algorithm
Factorial	O(n!)	Brute force travelling salesman problem

A theoretical measure of the execution of an algorithm, usually the time or memory needed, given the problem size n, which is usually the number of items. ... Formal Definition: f(n) = O(g(n)) means there are positive constants c and k, such that 0 ≤ f(n) ≤ cg(n) for all n ≥ k.

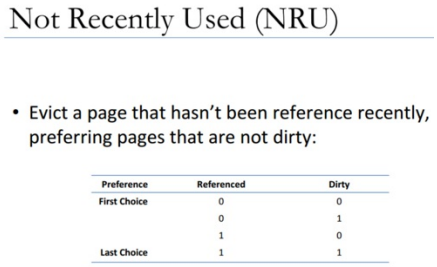
4. **Binary Search**



Search a sorted array by repeatedly dividing the search interval in half. Begin with an interval covering the whole array. If the value of the search key is less than the item in the middle of the interval, narrow the interval to the lower half. Otherwise narrow it to the upper half. Repeatedly check until the value is found or the interval is empty.

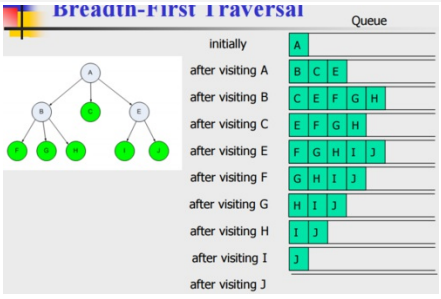
The complexity of binary search is O(logn)

5. **Bit Algorithms**



Bit manipulation is the act of algorithmically manipulating bits or other pieces of data shorter than a word. Computer programming tasks that require bit manipulation include low-level device control, error detection and correction algorithms, data compression, encryption algorithms, and optimization.

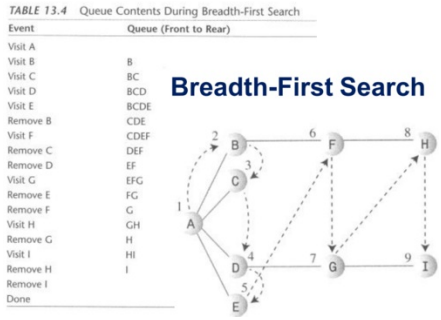
6. **Branch and Bound Algorithms**



Branch and bound (BB, B&B, or BnB) is an algorithm design paradigm for discrete and combinatoric optimization problems, as well as mathematical optimization.

Branch and bound is a technique used in integer optimization problems - ie optimization problems for which the variables are integers. Often (though not always) the variables are constrained to have values of either 0 or 1.

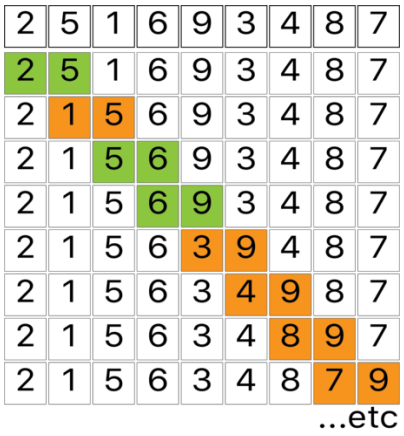
7. **Breadth First Search**



Breadth-first search (BFS) is an algorithm for traversing or searching tree or graph data structures. It starts at the tree root (or some arbitrary node of a graph, sometimes referred to as a 'search key') and explores the neighbor nodes first, before moving to the next level neighbors.

So for V number of vertices time complexity becomes O(V*N) = O(E) , where E is the total number of edges in the graph. Since removing and adding a vertex from/to Queue is O(1) , why it is added to the overall time complexity of BFS as O(V+E)

8. **Bubble Sort**



Bubble Sort is the simplest sorting algorithm that works by repeatedly swapping the adjacent elements if they are in wrong order.

Worst and Average Case Time Complexity: O(n*n). Worst case occurs when array is reverse sorted.

Best Case Time Complexity: O(n). Best case occurs when array is already sorted.

Auxiliary Space: O(1)

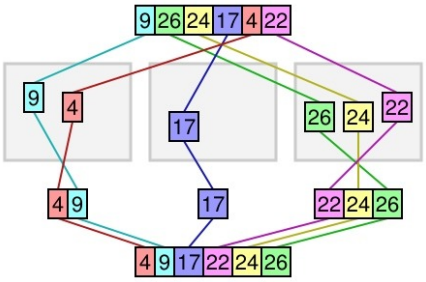
Boundary Cases: Bubble sort takes minimum time (Order of n) when elements are already sorted.

Sorting In Place: Yes

Stable: Yes

Due to its simplicity, bubble sort is often used to introduce the concept of a sorting algorithm.
In computer graphics it is popular for its capability to detect a very small error (like swap of just two elements) in almost-sorted arrays and fix it with just linear complexity (2n).

9. **Bucket Sort**

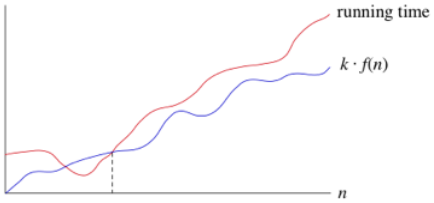


Bucket sort, or bin sort, is a sorting algorithm that works by distributing the elements of an array into a number of buckets. Each bucket is then sorted individually, either using a different sorting algorithm, or by recursively applying the bucket sorting algorithm.

Time Complexity: The complexity of bucket sort isn't constant depending on the input. However in the average case the complexity of the algorithm is O(n + k) where n is the length of the input sequence, while k is the number of buckets. The problem is that its worst-case performance is O(n^2) which makes it as slow as bubble sort.

USED WHEN: you can guarantee that your input is approximately uniformly distributed over a range.

10. **Complexity Spaces**

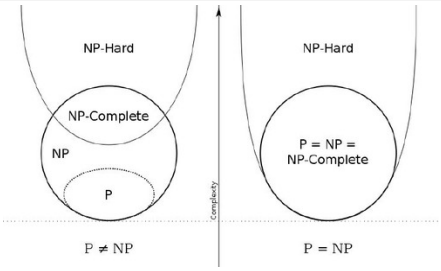


In computer science, the time complexity of an algorithm quantifies the amount of time taken by an algorithm to run as a function of the length of the string representing the input. The time complexity of an algorithm is commonly expressed using big O notation, which excludes coefficients and lower order terms.

Time complexity is commonly estimated by counting the number of elementary operations performed by the algorithm, where an elementary operation takes a fixed amount of time to perform. Thus, the amount of time taken and the number of elementary operations performed by the algorithm differ by at most a constant factor.

Since an algorithm's performance time may vary with different inputs of the same size, one commonly uses the worst-case time complexity of an algorithm, denoted as T(n), which is defined as the maximum amount of time taken on any input of size n.

11. **Complexity Theory**



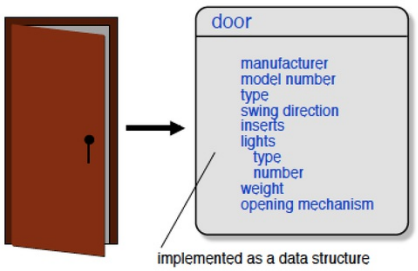
Computational complexity theory is a branch of the theory of computation in theoretical computer science that focuses on classifying computational problems according to their inherent difficulty, and relating those classes to each other.

12. **Computability Theory (Recursion Theory)**

Computability is the ability to solve a problem in an effective manner. It is a key topic of the field of computability theory within mathematical logic and the theory of computation within computer science.

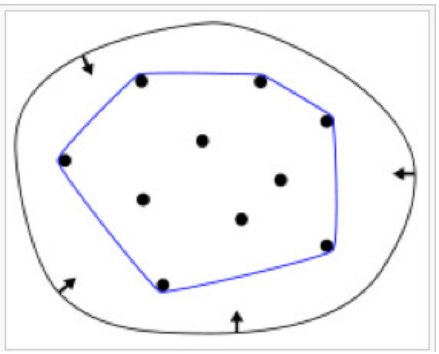
The basic questions addressed by recursion theory are "What does it mean for a function on the natural numbers to be computable?" and "How can noncomputable functions be classified into a hierarchy based on their level of noncomputability?". The answers to these questions have led to a rich theory that is still being actively researched.

13. **Concrete vs. Abstract Data**



To put it simple, ADT is a logical description and data structure is concrete. ADT is the logical picture of the data and the operations to manipulate the component elements of the data. Data structure is the actual representation of the data during the implementation and the algorithms to manipulate the data elements.

14. **Convex Hull Algorithm**



GEOMETRIC ALGORITHM:

Algorithms that construct convex hulls of various objects have a broad range of applications in mathematics and computer science.

In computational geometry, numerous algorithms are proposed for computing the convex hull of a finite set of points, with various computational complexities.

Computing the convex hull means that a non-ambiguous and efficient representation of the required convex shape is constructed. The complexity of the corresponding algorithms is usually estimated in terms of n, the number of input points, and h, the number of points on the convex hull.

Graham scan or another convex hull algorithm, for problems such as building a minimal fence to enclose animals.

15. **Counting Sort**

Counting sort is a sorting technique based on keys between a specific range. It works by counting the number of objects having distinct key values (kind of hashing). Then doing some arithmetic to calculate the position of each object in the output sequence.

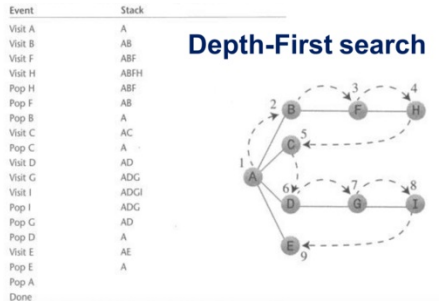
Time Complexity: O(n+k) where n is the number of elements in input array and k is the range of input.

Auxiliary Space: O(n+k)

Points to be noted:
1. Counting sort is efficient if the range of input data is not significantly greater than the number of objects to be sorted. Consider the situation where the input sequence is between range 1 to 10K and the data is 10, 5, 10K, 5K.
2. It is not a comparison based sorting. It running time complexity is O(n) with space proportional to the range of data.
3. It is often used as a sub-routine to another sorting algorithm like radix sort.
4. Counting sort uses a partial hashing to count the occurrence of the data object in O(1).
5. Counting sort can be extended to work for negative inputs also.

USED WHEN: you are sorting integers with a limited range.

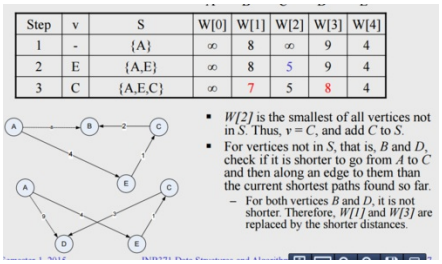
16. **Depth First Search**



Depth-first search (DFS) is an algorithm for traversing or searching tree or graph data structures. One starts at the root (selecting some arbitrary node as the root in the case of a graph) and explores as far as possible along each branch before backtracking.

The time complexity for DFS is O(n + m). We get this complexity considering the fact that we are visiting each node only once and in the case of a tree (no cycles) we are crossing all the edges once.

17. **Dijkstra's Algorithm**



Step	v	S	W[0]	W[1]	W[2]	W[3]	W[4]
1	-	{A}	∞	8	∞	9	4
2	E	{A,E}	∞	8	5	9	4
3	C	{A,E,C}	∞	7	5	8	4

- *W[2]* is the smallest of all vertices not in *S*. Thus, $v = C$, and add *C* to *S*.
- For vertices not in *S*, that is, *B* and *D*, check if it is shorter to go from *A* to *C* and then along an edge to them than the current shortest paths found so far.
  - For both vertices *B* and *D*, it is not shorter. Therefore, *W[1]* and *W[3]* are replaced by the shorter distances.

SHORTEST PATH ALGORITHM:

Dijkstra's algorithm is an algorithm for finding the shortest paths between nodes in a graph.

The algorithm exists in many variants; Dijkstra's original variant found the shortest path between two nodes, but a more common variant fixes a single node as the "source" node and finds shortest paths from the source to all other nodes in the graph, producing a shortest-path tree.

For a given source node in the graph, the algorithm finds the shortest path between that node and every other. It can also be used for finding the shortest paths from a single node to a single destination node by stopping the algorithm once the shortest path to the destination node has been determined.

Whenever you have a cost minimization problem with a (reasonably small) finite number of states, an initial state a target state, you can look at it as a pathfinding problem.

18. **Divide and Conquer Algorithms**

$$T(n) = 3T(n/2) + n.$$

In computer science, divide and conquer (D&C) is an algorithm design paradigm based on multi-branched recursion. A divide and conquer algorithm works by recursively breaking down a problem into two or more sub-problems of the same or related type, until these become simple enough to be solved directly.

19. **Dynamic Programming**

TABLE 9.2

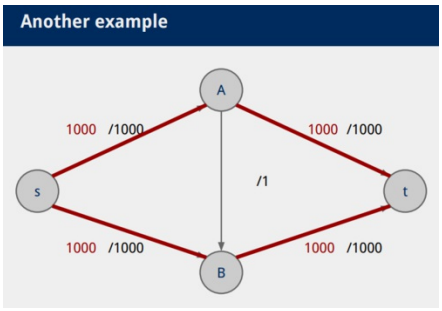
Core Training Parameters

Variables	Exercise Selection

• Plane of motion
  • Sagittal
  • Frontal
  • Transverse
• Range of motion
  • Full
  • Partial
  • End-range
• Type of resistance
  • Cable
  • Tubing
  • Medicine ball
  • Power ball
  • Dumbbells
  • Kettlebells
• Body position
  • Supine
  • Prone
  • Side-lying
  • Kneeling
  • Half-kneeling
  • Standing
  • Staggered-stance
  • Single-leg
  • Standing progression on unstable surface
• Speed of motion
  • Stabilization
  • Strength
  • Power
• Duration
• Frequency
• Amount of feedback
  • Fitness Professional's cues
  • Kinesthetic awareness
	• Progressive
	  • Easy to hard
	  • Simple to complex
	  • Known to unknown
	  • Stable to unstable
	• Systematic
	  • Stabilization
	  • Strength
	  • Power
	• Activity/Goal-specific
	• Integrated
	• Proprioceptively challenging
	  • Stability ball
	  • BOSU
	  • Reebok Core Board
	  • Half foam roll
	  • Airex pad
	  • BodyBlade
	• Based in current science

In computer science, mathematics, management science, economics and bioinformatics, dynamic programming (also known as dynamic optimization) is a method for solving a complex problem by breaking it down into a collection of simpler subproblems, solving each of those subproblems just once, and storing their solutions.

20. **Edmonds-Karp**

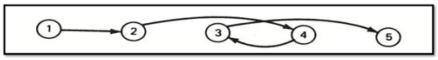


PATH (MAX FLOW) ALGORITHM:

In computer science, the Edmonds-Karp algorithm is an implementation of the Ford-Fulkerson method for computing the maximum flow in a flow network in O(V E2) time. The algorithm is identical to the Ford-Fulkerson algorithm, except that the search order when finding the augmenting path is defined. The path found must be a shortest path that has available capacity. This can be found by a breadth-first search, as we let edges have unit length. The running time of O(V E2) is found by showing that each augmenting path can be found in O(E) time, that every time at least one of the E edges becomes saturated (an edge which has the maximum possible flow), that the distance from the saturated edge to the source along the augmenting path must be longer than last time it was saturated, and that the length is at most V. Another property of this algorithm is that the length of the shortest augmenting path increases monotonically.

Edmonds-Karp for max flow/min cut problems. One common application is bipartite matching problems. For example, given N people, M food items, and a list of each person's food allergies, how many people can you feed?

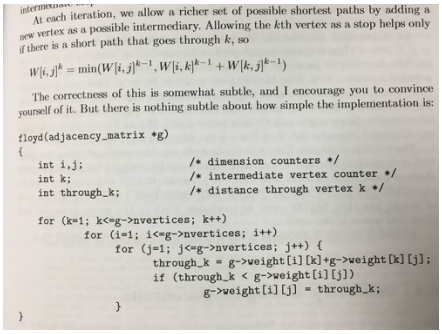
21. **File System Traversal and Manipulation**



A path traversal attack (also known as directory traversal) aims to access files and directories that are stored outside the web root folder. By manipulating variables that reference files with "dot-dot-slash (../)" sequences and its variations or by using absolute file paths, it may be possible to access arbitrary files and directories stored on file system including application source code or configuration and critical system files. It should be noted that access to files is limited by system operational access control (such as in the case of locked or in-use files on the Microsoft Windows operating system).

This attack is also known as "dot-dot-slash", "directory traversal", "directory climbing" and "backtracking".

22. **Floyd-Warshall**



At each iteration, we allow a richer set of possible shortest paths by adding a new vertex as a possible intermediary. Allowing the kth vertex as a stop helps only if there is a short path that goes through k, so

$$W[i,j]^k = \min(W[i,j]^{k-1}, W[i,k]^{k-1} + W[k,j]^{k-1})$$

The correctness of this is somewhat subtle, and I encourage you to convince yourself of it. But there is nothing subtle about how simple the implementation is:

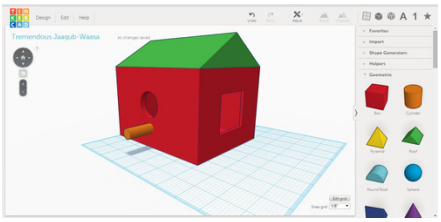
```
floyd(adjacency_matrix *g)
{
    int i,j;                    /* dimension counters */
    int k;                      /* intermediate vertex counter */
    int through_k;              /* distance through vertex k */

    for (k=1; k<=g->nvertices; k++)
        for (i=1; i<=g->nvertices; i++)
            for (j=1; j<=g->nvertices; j++) {
                through_k = g->weight[i][k]+g->weight[k][j];
                if (through_k < g->weight[i][j])
                    g->weight[i][j] = through_k;
            }
}
```

SHORTEST PATH ALGORITHM:

In computer science, the Floyd-Warshall algorithm is an algorithm for finding shortest paths in a weighted graph with positive or negative edge weights (but with no negative cycles). A single execution of the algorithm will find the lengths (summed weights) of the shortest paths between all pairs of vertices.

Floyd-Warshall is useful for computing all paths. It is sometimes used in problems where you don't need all paths, because it's so easy to implement. It is slower than other pathfinding algorithms though, so whether Floyd-Warshall is an option depends on the graph size.

23. **Geometric Algorithms**



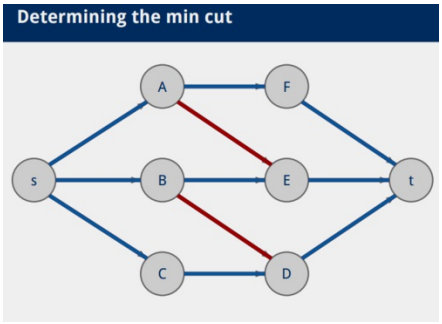
Computational geometry is a branch of computer science devoted to the study of algorithms which can be stated in terms of geometry. Some purely geometrical problems arise out of the study of computational geometric algorithms, and such problems are also considered to be part of computational geometry.

The main branches of computational geometry are:

Combinatorial computational geometry, also called algorithmic geometry, which deals with geometric objects as discrete entities.

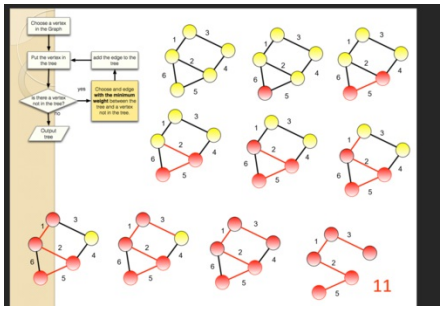
Numerical computational geometry, also called machine geometry, computer-aided geometric design (CAGD), or geometric modeling, which deals primarily with representing real-world objects in forms suitable for computer computations in CAD/CAM systems.

24. **Graph Algorithms**



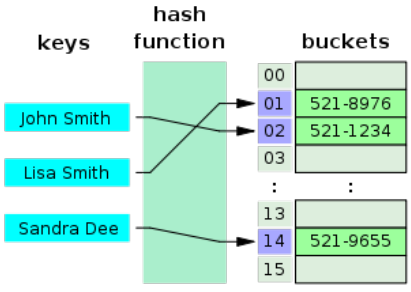
A graph algorithm is an algorithm that takes one or more graphs as inputs.
Performance constraints on graph algorithms are generally expressed in terms of the number of vertices (IV I) and the number of edges (IEI) in the input graph.

25. **Greedy Algorithms**



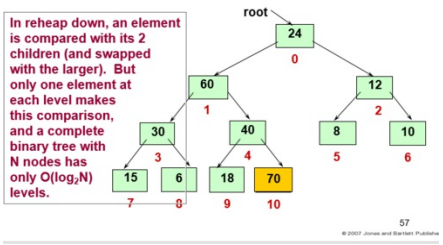
A greedy algorithm is an algorithmic paradigm that follows the problem solving heuristic of making the locally optimal choice at each stage with the hope of finding a global optimum.

26. **Hash Functions**



A hash function is any function that can be used to map data of arbitrary size to data of fixed size. The values returned by a hash function are called hash values, hash codes, digests, or simply hashes. One use is a data structure called a hash table, widely used in computer software for rapid data lookup.

27. **Heap Sort**



Heap sort is a comparison based sorting technique based on Binary Heap data structure. It is similar to selection sort where we first find the maximum element and place the maximum element at the end. We repeat the same process for remaining element.

Binary Heap DS:
10(0)
/ \
5(1) 3(2)
/ \
4(3) 1(4)

Notes:
Heap sort is an in-place algorithm.
Its typical implementation is not stable, but can be made stable.
Heap sort algorithm has limited uses because Quicksort and Mergesort are better in practice. Nevertheless, the Heap data structure itself is enormously used.

Time Complexity: Time complexity of heapify is O(Logn). Time complexity of createAndBuildHeap() is O(n) and overall time complexity of Heap Sort is O(nLogn).

USED WHEN: you don't need a stable sort and you care more about worst case performance than average case performance. It's guaranteed to be O(N log N), and uses O(1) auxiliary space, meaning that you won't unexpectedly run out of heap or stack space on very large inputs.

28. **Hungarian algorithm**

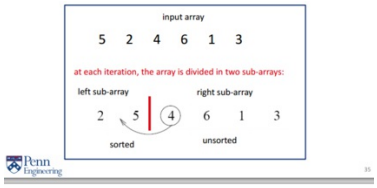
		Worker			
		A	B	C	D
Job	1	8	6	2	4
	2	6	7	11	10
	3	3	5	7	6
	4	5	10	12	9

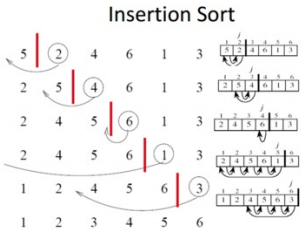
PATH ALGORITHM:

The Hungarian method is a combinatorial optimization algorithm that solves the assignment problem in polynomial time and which anticipated later primal-dual methods.

The Hungarian algorithm is for assignment problems. Similar to the above, but in these problems the edges have weights, and we're maximizing the total weight rather than just the number of matchings.

29. **Insertion Sort**





Insertion sort is a simple sorting algorithm that builds the final sorted array (or list) one item at a time.

Time Complexity: O(n*n)

Auxiliary Space: O(1)

Algorithmic Paradigm: Incremental Approach

Boundary Cases: Insertion sort takes maximum time to sort if elements are sorted in reverse order. And it takes minimum time (Order of n) when elements are already sorted.

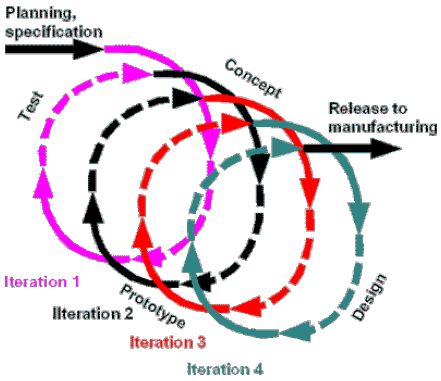
Sorting In Place: Yes

Stable: Yes

Online: Yes

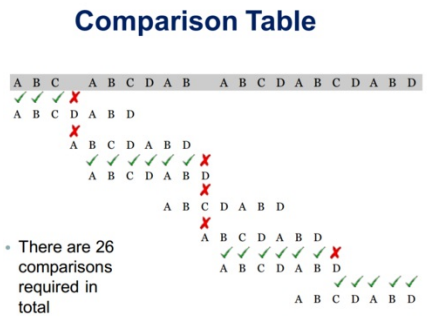
USED WHEN: Insertion sort is used when number of elements is small. It can also be useful when input array is almost sorted, only few elements are misplaced in complete big array.

30. **Iteration**



In computational mathematics, an iterative method is a mathematical procedure that generates a sequence of improving approximate solutions for a class of problems, in which the n-th approximation is derived from the previous ones.

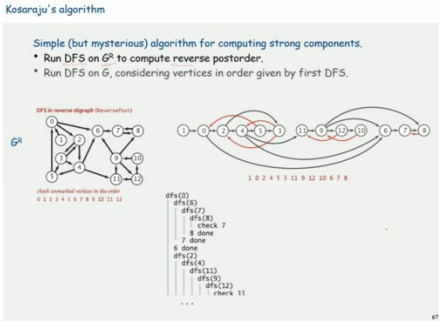
31. **Knuth-Morris-Pratt Algorithm**



STRING SEARCHING:

In computer science, the Knuth-Morris-Pratt string searching algorithm (or KMP algorithm) searches for occurrences of a "word" W within a main "text string" S by employing the observation that when a mismatch occurs, the word itself embodies sufficient information to determine where the next match could begin, thus bypassing re-examination of previously matched characters.

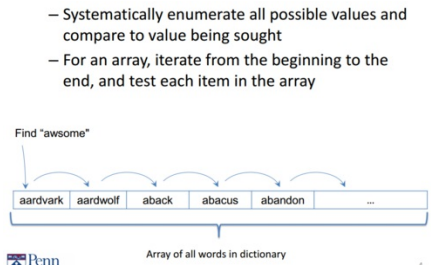
32. **Kosaraju's Algorithm**



STRONGLY CONNECTED COMPONENT ALGORITHM:

Kosaraju's algorithm uses two passes of depth first search. The first, in the original graph, is used to choose the order in which the outer loop of the second depth first search tests vertices for having been visited already and recursively explores them if not. The second depth first search is on the transpose graph of the original graph, and each recursive exploration finds a single new strongly connected component.

33. **Linear Search**

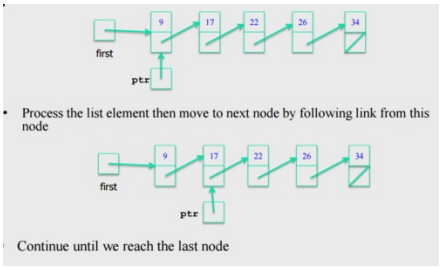


In computer science, linear search or sequential search is a method for finding a target value within a list. It sequentially checks each element of the list for the target value until a match is found or until all the elements have been searched.

Linear search is rarely used practically because other search algorithms such as the binary search algorithm and hash tables allow significantly faster searching comparison to Linear search.

The worst case complexity of linear search is O(n)

34. **List Traversal Algorithms**



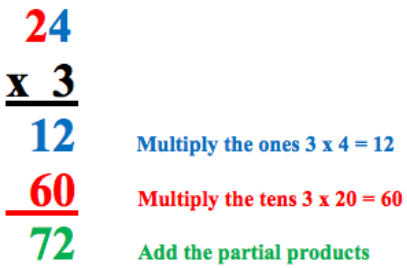
Assume, that we have a list with some nodes. Traversal is the very basic operation, which presents as a part in almost every operation on a singly-linked list. For instance, algorithm may traverse a singly-linked list to find a value, find a position for insertion, etc. For a singly-linked list, only forward direction traversal is possible.

Traversal algorithm

Beginning from the head,

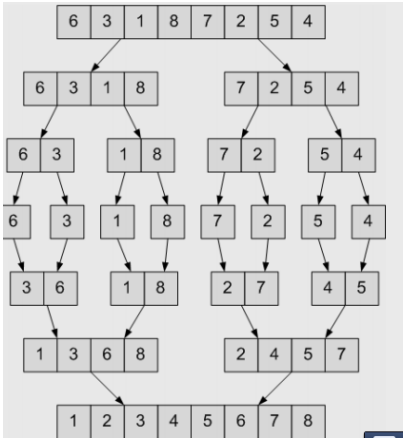
1.) check, if the end of a list hasn't been reached yet;
2.) do some actions with the current node, which is specific for particular algorithm;
3.) current node becomes previous and next node becomes current. Go to the step 1.

35. **Mathematical Algorithms**



An algorithm is a list of well-defined instructions or a step-by-step procedure to solve a problem

36. **Merge Sort**



Merge Sort is a Divide and Conquer algorithm. It divides input array in two halves, calls itself for the two halves and then merges the two sorted halves. The merge() function is used for merging two halves. The merge(arr, l, m, r) is key process that assumes that arr[l..m] and arr[m+1..r] are sorted and merges the two sorted sub-arrays into one.

Time Complexity: . \Theta(nLogn) in all 3 cases (worst, average and best) as merge sort always divides the array in two halves and take linear time to merge two halves.

Auxiliary Space: O(n)

Algorithmic Paradigm: Divide and Conquer

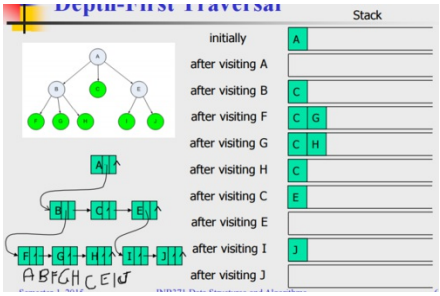
Sorting In Place: No in a typical implementation

Stable: Yes

Merge Sort is useful for sorting linked lists in O(nLogn) time.In case of linked lists the case is different mainly due to difference in memory allocation of arrays and linked lists. Unlike arrays, linked list nodes may not be adjacent in memory. Unlike array, in linked list, we can insert items in the middle in O(1) extra space and O(1) time. Therefore merge operation of merge sort can be implemented without extra space for linked lists.

USED WHEN: you need a stable, O(N log N) sort (this is about your only option). The only downsides to it are that it uses O(N) auxiliary space and has a slightly larger constant than a quick sort. There are some in-place merge sorts, but AFAIK they are all either not stable or worse than O(N log N). Even the O(N log N) in place sorts have so much larger a constant than the plain old merge sort that they're more theoretical curiosities than useful algorithms.

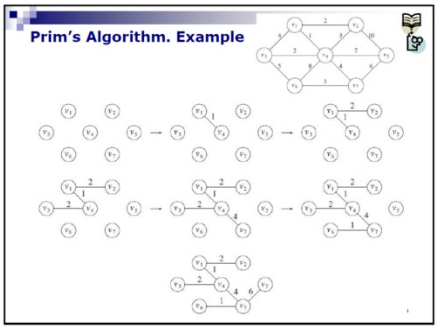
37. **Path-Based Strong Component Algorithm**



STRONGLY CONNECTED COMPONENT ALGORITHM:

The path-based strong component algorithm uses a depth first search, like Tarjan's algorithm, but with two stacks. One of the stacks is used to keep track of the vertices not yet assigned to components, while the other keeps track of the current path in the depth first search tree.

38. **Prim's Algorithm**



MINIMUM SPANNING TREE:

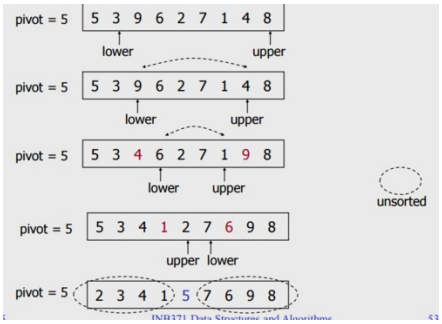
In computer science, Prim's algorithm is a greedy algorithm that finds a minimum spanning tree for a weighted undirected graph. This means it finds a subset of the edges that forms a tree that includes every vertex, where the total weight of all the edges in the tree is minimized.

39. **Quantum Bogo Sort**

Quantum Bogo Sort a quantum sorting algorithm which can sort any list in O(1), using the "many worlds" interpretation of quantum mechanics.

It works as follows:
1.) Quantumly randomise the list, such that there is no way of knowing what order the list is in until it is observed. This will divide the universe into O(n!) universes; however, the division has no cost, as it happens constantly anyway.
2.) If the list is not sorted, destroy the universe. (This operation is left as an exercise to the reader.)
3.) All remaining universes contain lists which are sorted.

40. **Quick Sort**



QuickSort is a Divide and Conquer algorithm. It picks an element as pivot and partitions the given array around the picked pivot. There are many different versions of quickSort that pick pivot in different ways.

e.g.
1.) Always pick first element as pivot.
2.) Always pick last element as pivot
3.) Pick a random element as pivot.
4.) Pick median as pivot.

The key process in quickSort is partition(). Target of partitions is, given an array and an element x of array as pivot, put x at its correct position in sorted array and put all smaller elements (smaller than x) before x, and put all greater elements (greater than x) after x. All this should be done in linear time.

Time complexity: \theta(n2) (worst case), \theta(nLogn) (best case)

Quick Sort in its general form is an in-place sort (i.e. it doesn't require any extra storage) whereas merge sort requires O(N) extra storage, N denoting the array size which may be quite expensive. Allocating and de-allocating the extra space used for merge sort increases the running time of the algorithm. Comparing average complexity we find that both type of sorts have O(NlogN) average complexity but the constants differ. For arrays, merge sort loses due to the use of extra O(N) storage space.

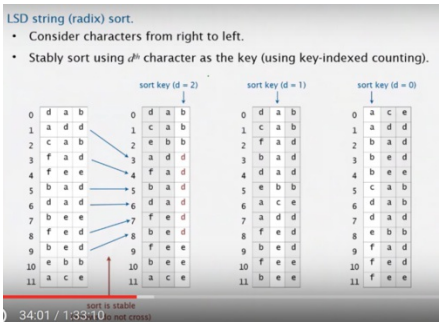
Most practical implementations of Quick Sort use randomized version. The randomized version has expected time complexity of O(nLogn). The worst case is possible in randomized version also, but worst case doesn't occur for a particular pattern (like sorted array) and randomized Quick Sort works well in practice.

Quick Sort is also a cache friendly sorting algorithm as it has good locality of reference when used for arrays.

Quick Sort is also tail recursive, therefore tail call optimizations is done.

USED WHEN: you don't need a stable sort and average case performance matters more than worst case performance. A quick sort is O(N log N) on average, O(N^2) in the worst case. A good implementation uses O(log N) auxiliary storage in the form of stack space for recursion.

41. **Radix Sort**



In computer science, radix sort is a non-comparative integer sorting algorithm that sorts data with integer keys by grouping keys by the individual digits which share the same significant position and value.

The lower bound for Comparison based sorting algorithm (Merge Sort, Heap Sort, Quick-Sort .. etc) is Ω(nLogn), i.e., they cannot do better than nLogn.

Time Complexity: Ω(nk)
Space Complexity: O(n+k)
... when elements are in range from 1 to k.

USED WHEN: log(N) is significantly larger than K, where K is the number of radix digits

42. **Randomized Algorithms**

```
vector<int> ComputeRandomPermutation(int n) {
  vector<int> permutation(n);
  // Initializes permutation to 0, 1, 2, ..., n - 1.
  iota(permutation.begin(), permutation.end(), 0);
  RandomSampling(permutation.size(), &permutation);
  return permutation;
}
```

A randomized algorithm is an algorithm that employs a degree of randomness as part of its logic. The algorithm typically uses uniformly random bits as an auxiliary input to guide its behavior, in the hope of achieving good performance in the "average case" over all possible choices of random bits.

43. **Scheduling Algorithms**

Job	CPU (Burst Time)
1	24
2	3
3	3

With the SJF algorithm we get the following order

P2	P3	P1

SJF gives the following performance results:

Job	Waiting Time	Response Time	Turnaround Time
1	6	6	30
2	0	0	3
3	3	3	6
Average	3	3	13

A scheduling algorithm is the algorithm which dictates how much CPU time is allocated to Processes and Threads. The goal of any scheduling algorithm is to fulfill a number of criteria:

1.) No task must be starved of resources - all tasks must get their chance at CPU time;

2.) If using priorities, a low-priority task must not hold up a high-priority task;

3.) The scheduler must scale well with a growing number of tasks, ideally being O(1). This has been done, for example, in the Linux kernel.

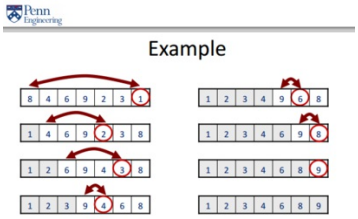
44. **Searching Algorithms**

```
/**
 Returns true if item is on the list;
 otherwise, returns false. Does not differentiate
 between uppercase and lowercase letters.
*/
public boolean isOnList(String item)
{
    boolean found = false;
    int i = 0;
    while (!found && (i < countOfEntries))
    {
        if (item.equalsIgnoreCase(entry[i]))
            found = true;
        else
            i++;
    }
    return found;
}
```

A search algorithm is the step-by-step procedure used to locate specific data among a collection of data. It is considered a fundamental procedure in computing. In computer science, when searching for data, the difference between a fast application and a slower one often lies in the use of the proper search algorithm.

45. **Selection Sort**

- Idea:
  - Find the smallest element in the array
  - Exchange it with the element in the first position
  - Find the second smallest element and exchange it with the element in the second position
  - Continue until the array is sorted



The selection sort algorithm sorts an array by repeatedly finding the minimum element (considering ascending order) from unsorted part and putting it at the beginning. The algorithm maintains two subarrays in a given array.

1) The subarray which is already sorted.
2) Remaining subarray which is unsorted.

In every iteration of selection sort, the minimum element (considering ascending order) from the unsorted subarray is picked and moved to the sorted subarray.

Time Complexity: O(n2) as there are two nested loops.

Auxiliary Space: O(1)

The good thing about selection sort is it never makes more than O(n) swaps and can be useful when memory write is a costly operation.

USED WHEN: When you're doing something quick and dirty and for some reason you can't just use the standard library's sorting algorithm. The only advantage these have over insertion sort is being slightly easier to implement.

46. **Sorting Algorithms**

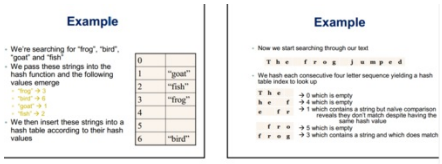
```
/* use insertion sort to sort x[0], x[1], …, x[n-1] into ascending order*/
for i = 1 to n-1 do
    /* insert x[i] into its proper position among x[0], x[1], … , x[i-1] */
    temp = x[i];
    j = i-1;
    while (j>=0) and (temp < x[j]) do
        x[j+1] = x[j];
        j = j – 1;
    end while
    x[j+1] = temp;
end for
```

In computer science a sorting algorithm is an algorithm that puts elements of a list in a certain order. The most-used orders are numerical order and lexicographical order.

47. **String Manipulation**

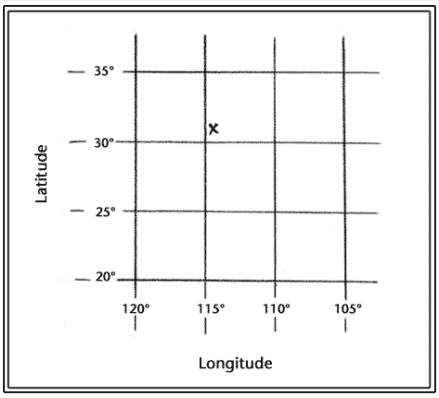
Most programming languages that have a string datatype will have some string functions although there may be other low-level ways within each language to handle strings directly. In object-oriented languages, string functions are often implemented as properties and methods of string objects. In functional and list-based languages a string is represented as a list (of character codes), therefore all list-manipulation procedures could be considered string functions. However such languages may implement a subset of explicit string-specific functions as well.

48. **String Searching Algorithms**



In computer science, string searching algorithms, sometimes called string matching algorithms, are an important class of string algorithms that try to find a place where one or several strings (also called patterns) are found within a larger string or text.

49. **Sweep Line "Algorithm"**



GEOMETRIC ALGORITHM:

In computational geometry, a sweep line algorithm or plane sweep algorithm is an algorithmic paradigm that uses a conceptual sweep line or sweep surface to solve various problems in Euclidean space. It is one of the key techniques in computational geometry.

The sweep line "algorithm" (more of a general approach really) is useful for various geometric problems, like the nearest pair problem. Also useful for a variety of intersection-related problems, like finding intersecting line segments, or conflicting calendar events.

50. **Tarjan's Algorithm**

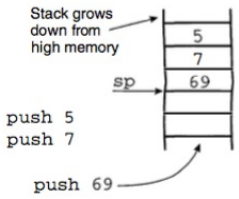
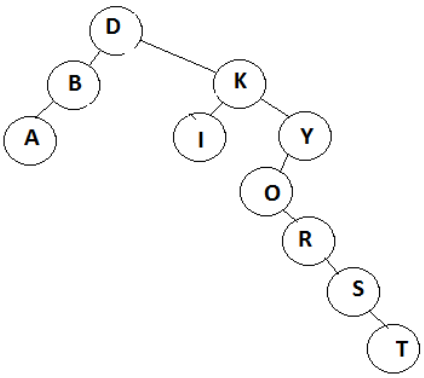


Figure 8.1 Stack data structure.

STRONGLY CONNECTED COMPONENT ALGORITHM:

Tarjan's strongly connected components algorithm performs a single pass of depth first search. It maintains a stack of vertices that have been explored by the search but not yet assigned to a component, and calculates "low numbers" of each vertex (an index number of the highest ancestor reachable in one step from a descendant of the vertex) which it uses to determine when a set of vertices should be popped off the stack into a new component.

51. **Tree Traversal Algorithms**



In computer science, tree traversal (also known as tree search) is a form of graph traversal and refers to the process of visiting (checking and/or updating) each node in a tree data structure, exactly once. ... The following algorithms are described for a binary tree, but they may be generalized to other trees as well.

52. **Types of Search Algorithms**

Linear Search,
Binary Search,
Depth First Search,
Breadth First Search,

53. **Types of Sorting Algorithms**

Bubble Sort,
Insertion Sort,
Merge Sort,
Quick Sort,
Radix style sorts, Counting Sort and Bucket Sort (NON-COMPARISON SORTS),
Heap Sort,
Bogo and Quantum Sort,
Selection Sort,