Stack

<https://www.geeksforgeeks.org/design-and-implement-special-stack-data-structure/>

<https://www.geeksforgeeks.org/design-a-stack-that-supports-getmin-in-o1-time-and-o1-extra-space/> // See once

<https://www.geeksforgeeks.org/implement-stack-using-priority-queue-or-heap/> // Not done

<https://www.geeksforgeeks.org/efficiently-implement-k-stacks-single-array/> // IMP

<https://www.geeksforgeeks.org/next-greater-element/>

<https://www.geeksforgeeks.org/the-celebrity-problem/> //Important

<https://www.geeksforgeeks.org/iterative-tower-of-hanoi/>

<https://www.geeksforgeeks.org/sort-a-stack-using-recursion/>

<https://www.geeksforgeeks.org/sort-stack-using-temporary-stack/>

<https://www.geeksforgeeks.org/merging-intervals/>

<https://www.geeksforgeeks.org/largest-rectangle-under-histogram/>

<https://www.geeksforgeeks.org/find-the-maximum-of-minimums-for-every-window-size-in-a-given-array/>

<https://www.geeksforgeeks.org/remove-brackets-algebraic-string-containing-operators/>

<https://www.geeksforgeeks.org/check-mirror-n-ary-tree/>

<https://www.geeksforgeeks.org/check-for-balanced-parenthesis-without-using-stack/>

<https://www.geeksforgeeks.org/minimum-number-of-bracket-reversals-needed-to-make-an-expression-balanced/>

<https://www.geeksforgeeks.org/expression-contains-redundant-bracket-not/>

<https://www.geeksforgeeks.org/find-next-smaller-next-greater-array/>

<https://www.geeksforgeeks.org/check-two-expressions-brackets/>

Queue

<https://www.geeksforgeeks.org/implement-a-stack-using-single-queue/>

<https://www.geeksforgeeks.org/efficiently-implement-k-queues-single-array/>

<https://www.geeksforgeeks.org/lru-cache-implementation/>

<https://www.geeksforgeeks.org/stack-permutations-check-if-an-array-is-stack-permutation-of-other/>

<https://www.geeksforgeeks.org/sliding-window-maximum-maximum-of-all-subarrays-of-size-k/>

<https://www.geeksforgeeks.org/find-the-largest-number-multiple-of-3/>

<https://www.geeksforgeeks.org/reversing-first-k-elements-queue/>

<https://www.geeksforgeeks.org/interleave-first-half-queue-second-half/>

<https://www.geeksforgeeks.org/number-siblings-given-node-n-ary-tree/>

<https://www.geeksforgeeks.org/check-if-a-given-binary-tree-is-complete-tree-or-not/>

<https://www.geeksforgeeks.org/given-linked-list-representation-of-complete-tree-convert-it-to-linked-representation/>

<https://www.geeksforgeeks.org/check-queue-can-sorted-another-queue-using-stack/>

<https://www.geeksforgeeks.org/implement-a-stack-using-single-queue/>

<https://www.geeksforgeeks.org/implement-stack-queue-using-deque/>

<https://www.geeksforgeeks.org/minimum-time-required-so-that-all-oranges-become-rotten/>

<https://www.geeksforgeeks.org/distance-nearest-cell-1-binary-matrix/>

1. Abhishek’s Approach- make 1s - zeros and 0s to ones. Star from bottom right most cell, if it’s value is 1 then look for min in all four direction. m[i][j]= min +1. Go up likewise till m[0][0].
2. DP Approach

f(matrix.size()==0)return {};

int n=matrix.size(),m=matrix[0].size();

vector<vector<int>> dp(n,vector<int> (m,INT\_MAX-1));

for(int i=0;i<n;i++)

{

for(int j=0;j<m;j++)

{

if(matrix[i][j]==1)

dp[i][j]=1;

else

{

if(j>0)

dp[i][j]=min(dp[i][j],dp[i][j-1]+1);

if(i>0)

dp[i][j]=min(dp[i][j],dp[i-1][j]+1);

}

}

}

for(int i=n-1;i>=0;i--)

{

for(int j=m-1;j>=0;j--)

{

if(matrix[i][j]==1)

continue;

else

{

if(j<m-1)

dp[i][j]=min(dp[i][j],dp[i][j+1]+1);

if(i<n-1)

dp[i][j]=min(dp[i][j],dp[i+1][j]+1);

}

}

}

Linked List

Singly

<https://www.geeksforgeeks.org/nth-node-from-the-end-of-a-linked-list/>

<https://www.geeksforgeeks.org/write-a-c-function-to-print-the-middle-of-the-linked-list/> //Method 3

<https://www.geeksforgeeks.org/detect-loop-in-a-linked-list/>

<https://www.geeksforgeeks.org/how-does-floyds-slow-and-fast-pointers-approach-work/>

1) When slow pointer **enters the loop, the fast pointer must be inside the loop**. Let fast pointer be distance k from slow.

2) Now if consider movements of slow and fast pointers, we can notice that distance between them (from slow to fast) **increase by one after every iteration.** After one iteration (of slow = next of slow and fast = next of next of fast), distance between slow and fast becomes k+1, after two iterations, k+2, and so on. When distance becomes n, they meet because they are moving in a cycle of length n.

<https://www.geeksforgeeks.org/find-length-of-loop-in-linked-list/>

<https://www.geeksforgeeks.org/detect-and-remove-loop-in-a-linked-list/> // finding beginning of loop

<https://www.geeksforgeeks.org/function-to-check-if-a-singly-linked-list-is-palindrome/>

<https://www.geeksforgeeks.org/check-linked-list-loop-palindrome-not/>

1. find beginning of loop

2. use linked list palindrome algo (using beginning node as end of list)

<https://www.geeksforgeeks.org/remove-occurrences-duplicates-sorted-linked-list/>

<https://www.geeksforgeeks.org/swap-nodes-in-a-linked-list-without-swapping-data/> // write clean code

<https://www.geeksforgeeks.org/pairwise-swap-elements-of-a-given-linked-list-by-changing-links/>

<https://www.geeksforgeeks.org/pairwise-swap-adjacent-nodes-of-a-linked-list-by-changing-pointers-set-2/>

<https://www.geeksforgeeks.org/intersection-of-two-sorted-linked-lists/>

<https://www.geeksforgeeks.org/write-a-function-to-get-the-intersection-point-of-two-linked-lists/>

<https://www.geeksforgeeks.org/segregate-even-and-odd-elements-in-a-linked-list/>

<https://www.geeksforgeeks.org/reverse-a-linked-list/>

<https://www.geeksforgeeks.org/flattening-a-linked-list/>

<https://www.geeksforgeeks.org/a-linked-list-with-next-and-arbit-pointer/> //Method 2 takes O(1) space // Not done

<https://www.geeksforgeeks.org/clone-linked-list-next-arbit-pointer-set-2/>

Circular

<https://www.geeksforgeeks.org/split-a-circular-linked-list-into-two-halves/>

<https://www.geeksforgeeks.org/convert-a-binary-tree-to-a-circular-doubly-link-list/> // see once

<https://www.geeksforgeeks.org/josephus-circle-using-circular-linked-list/>

<https://www.geeksforgeeks.org/josephus-problem-set-1-a-on-solution/>

<https://www.geeksforgeeks.org/josephus-problem-using-bit-magic/> // maths (https://www.youtube.com/watch?v=uCsD3ZGzMgE) // incremented by 2 if not power of 2

<https://www.geeksforgeeks.org/exchange-first-last-node-circular-linked-list/>

<https://www.geeksforgeeks.org/reverse-circular-linked-list/>

Doubly

<https://www.geeksforgeeks.org/reverse-a-doubly-linked-list/>

<https://www.geeksforgeeks.org/reverse-doubly-linked-list-using-recursion/>

<https://www.geeksforgeeks.org/swap-kth-node-from-beginning-with-kth-node-from-end-in-a-linked-list/>

<https://www.geeksforgeeks.org/merge-sort-for-doubly-linked-list/>

<https://www.geeksforgeeks.org/count-triplets-sorted-doubly-linked-list-whose-sum-equal-given-value-x/> // IMP

<https://www.geeksforgeeks.org/sort-k-sorted-doubly-linked-list/>

<https://www.geeksforgeeks.org/rotate-doubly-linked-list-n-nodes/>

<https://www.geeksforgeeks.org/reverse-doubly-linked-list-groups-given-size/>

Misc

<https://www.geeksforgeeks.org/given-only-a-pointer-to-a-node-to-be-deleted-in-a-singly-linked-list-how-do-you-delete-it/>