Popular sums

Sunday, 2 July 2023 6:03 PM

Fair distribution of chocolates among children:

Back-tracing:

```
classSolution{
  int ans = Integer.MAX_VALUE;
voidhelper(int[] cookies, int start, int k, int[] temp){
    if(start == cookies.length) {
      int max = 0;
      for(int c : temp)
        max = Math. max(max, c);
      ans = Math. min(ans, max);
      return;
    }
    for(inti = 0; i < k; i++) {
      temp[i] += cookies[start];
      helper(cookies, start + 1, k, temp);
      temp[i] -= cookies[start];
    }
  }
Public int distributeCookies(int[] cookies, int k){
    helper(cookies, 0, k, new int[k]);
    return ans;
  }
}
```

te-csi2.qa.paypal.com:21454

Problem	Link	Snippet		

Common substring between 2 strings	https://leetcode.com/problems/minimum-ascii-delete-sum-for-two-strings/solution/		DP	if(s1.charAt(i-1) == s2.charAt(j-1)){
				dp[i][j] = dp[i-1][j-1];
				}else{
				dp[i][j] = Math.min(
				dp[i-1][j] + s1.charAt(i-1),
				dp[i][j-1] + s2.charAt(j-1)
				);
				}

All back tracking logics	https://leetcode.com/problems/permutations/discuss/18239/A-general-approach-to-backtracking-questions-in-Java-(Subsets-Permutations-Combination-Sum-Palindrome-Partioning)		Back tracking	Permutation, combination, subsets
Word in Worddictionary	https://leetcode.com/problems/word-break/		DP	1. DP in one dimension 2. Recursive solution 3. DP in two dimension

Search in Rotated Sorted Array
 	
When duplicates are allowed
	https://leetcode.com/problems/search-in-rotated-sorted-array/discuss/154836/The-INF-and-INF-method-but-with-a-better-explanation-for-dummies-like-me
	
	https://leetcode.com/problem
		if(nums[left]<nums[mid] && target>nums[left]){
		if(target<nums[mid])
		right=mid-1;
		else if(target>nums[mid])
		left=mid+1;
		}
		}else if(nums[right]>nums[mid] &&
			Binary search	1. Take a comparator value which is initially nums[mid], if both the target and nums[mid] are in same side then keep it as it is.
				2. If its on other side if target is less than

	https://leetcode.com/problems/search-in-rotated-sorted-array-ii/submissions/ (return true or false)
	https://leetcode.com/problems/search-in-rotated-sorted-array/submissions/ (return index)
		else if(nums[right]>nums[mid] && target>nums[mid]){
		if(target>nums[right])
		right=mid-1;
		else if(target>nums[mid])
		left=mid+1;
		}
				target is less then make is -INF and vice versa
				3. Do normal binary search after that
				(Or)
				1. Find real min value which is actual rotations done using BS and then every time doing the BS use the rotation value % to determine
				One solution for all Rotated Array:
				My own approach of comparing with start and end values

Any min-max problem (Minimize the Maximum Difference of Pairs)	https://leetcode.com/problems/minimize-the-maximum-difference-of-pairs/discuss/3395750/JavaC%2B%2BPython-Binary-Search		Binary search	Think about binary search solution first
Change for coins problem	https://leetcode.com/problems/coin-change-ii/discuss/675182/JAVA-or-100-Faster-or-DP			1. Think of using dp[0] instead of adding +1

Move robot from top-left to right-bottom corner	https://leetcode.com/problems/unique-paths-ii/submissions/	if (dp[i - 1][j]>0 && dp[i][j - 1]>0)
		dp[i][j] = dp[i - 1][j]+dp[i][j - 1];
		else if (dp[i - 1][j] > 0)
		dp[i][j] = dp[i - 1][j];
			[1, 1, 1
			1, 2, 3
			1, 3,
			6]
				1. At each point see if there are two ways at the point. If so add the values from

Easy problem : Given integer 26 give it in excel format as Z	https://leetcode.com/problems/excel-sheet-column-title/	build.append((char)('A'+(columnNumber-1)%26));
		columnNumber = (columnNumber-1)/26;
			String Builder	1. Have to minus 1, as 'A' + will start with 0 but in reality it starts with 1.

Stack using 2 queues	https://leetcode.com/problems/implement-stack-using-queues/			1. 2 queues initialize 2. Push : given element add first in q2, remaining element from q1 add to q2, swap q1 and q2. 3. Other operation are now performed on q1.

Count number of 1's from 0 to n in	https://leetcode.com/problems/counting-bits/submissions/	ans[0] = 0;
		for(int i=1; i<=n; i++) {
		ans[i] = ans[i/2] + i%2;
		}
				1. Simple method is create a dp array, with arr[0]=0,
				2. Then arr[i] = arr[i/2]+i%2, because each 2 number is formed by previous set by adding 0 or 1

Finding common char in string	https://leetcode.com/problems/extra-characters-in-a-string/			1. This is different approach in DP, first take each char one by one 2. Check whether its available in the given set 3. Then append the char one by one

				4. Form all combinations and from top to down and then from down to top increment 5. Sure method to learn to find the common words
Cloning any type Linked list	https://leetcode.com/problems/copy-list-with-random-pointer/discuss/1842118/Clone-Node-linked-list-with-random-pointer-Hashmap	One of my solutions under rated and beautifully explained		1. Store node to node in an HashMap 2. Append the values and store in HashMap 3. Return the head of the hashmap
Integer Break - to get maximum product	https://leetcode.com/problems/integer-break/	https://leetcode.com/problems/integer-break/submissions/		1. Power of 3, when its multiplied depending on 3 the value gets maximum product 2. If remainder is 1 then multiply by 4 and incase of 2 then multiply by 2
Sweep line algorithm for all time related problems	https://leetcode.com/problems/number-of-flowers-in-full-bloom/discuss/1976821/Java-Sweep-Line-Algorithm	Add separately start time, person and end_time to a single priority queue. Retrive from PQ and do the math based on the event		1. See everything as individual event 2. Add Person to the PQ as event 1 with {time position, human, index} 3. Add starttime to the same PQ as event 0 4. Add endtime to the same PQ as event 2 5. Now iterate and increment the count when event 0, decrement when event 2 and add to result for event 1
Number of ways problem :: Number of Ways to Stay in the Same Place After Some Steps	https://leetcode.com/problems/number-of-ways-to-stay-in-the-same-place-after-some-steps/discuss/437700/Very-simple-and-easy-to-understand-java-solution	Using DP to find the number of ways problems,		1. Use 2D then try with 1D
Observation on binary tree :: Find whether give array of left and right child is valid binary tree	https://leetcode.com/problems/validate-binary-tree-nodes/submissions/	1. Understanding the binary tree 2. Check whether there is only one root node 3. BFS of the binary tree to find whether all nodes are reachable	Tree	1. For a root node there will not be any in node, other nodes have one in node 2. If any one node is visited twice then its not a binary tree
Observation on tree :: Unless you think of returning two values sum and count as array it looks hard	https://leetcode.com/problems/count-nodes-equal-to-average-of-subtree/		Tree	1. BFS through the tree 2. If root is null return empty array 3. Else Do a recursion of left and right 4. To find sum and count do sum=left[0]+right[0]+root.val 5. To find count =left[1]+right[1]+1 6. Return new int[]{sum, count};
Decode Ways :: How many ways 12 is mapped to	https://leetcode.com/submissions/detail/540366398/	public int numDecodings(String s) {	DP	1. This problem has sub problem where after

ways 12 is mapped to Alphabet -> AB & L	ons/detail/940566598/	s) {
		int n=s.length() ;
		int[] dp=new int[n+1] ;
		dp[n]=1 ;
		for(int i = n-1 ; i >= 0 ; i--)
		if(s.charAt(i)!='0')
		{
		dp[i] = dp[i+1] ;
		if(i < n-1 && (s.charAt(i)=='1' || s.charAt(i)=='2' && s.charAt(i+1)<'7'))
		dp[i]+=dp[i+2];
		}
		return dp[0];
		}
				problem where after spending sometime, you will understand it should come from bottom to top
				2. If in the dp if the value is 0 we have keep the value as is, if its any other value have to retain the previous value
				3. If the value is 1 or 2 and the previous value is 10 to 26 then we have to add the previous value count to current value
				4. Finally have to return the dp[0]

Corona virus spread problem :: You are given the root of a binary tree with **unique** values, and an integer start. At minute 0, an **infection** starts from the node with value start	https://leetcode.com/problems/amount-of-time-for-binary-tree-to-be-infected/?envType=daily-question&envId=2024-01-10		Tree 2 Graph	1. Think little out of box, 2. Convert the tree into graph and then start with the problem 3. Tree to Map of adjacent values linking together
Maximum profit in job scheduling ::				

Construct Binary Tree from inorder and level order		Private Tree buildTree(Map<Integer,Integer> levelorder, int[] inorder, int start, int end){
		Int index=start;
		for(int i=start;i<end;i++){
		if(levelorder.get(inorder[i]) <levelorder.get(inorder[index])){
		index=levelorder.get(inorder[i]);
		}
		}
		Tree root=new Tree(inorder[index]);
		root.left=buildTree(levelorder,inorder,0,index-1);
		root.right=buildTree(levelorder,inorder,index+1,end);
		Return root;
		}
				1. Left of the inorder is on the left of tree and vise versa
				2. Construct a map storing index of level order

Alphabets filling in a given string with '?' in lexographic order	Given - 'a???a'
	Output - 'abcda'
	https://leetcode.com/contest/biweekly-contest-126/problems/replace-question-marks-in-string-to-minimize-its-value/
		int idx=(prev+1)%26-1;
		Sort((char)('a'+((idx>=0)?idx:25)))
				1. Storing the alphabet sequence is the key, managing 'a' as 1 till 'z' as 26
				2. Store in map
				3. Get the less freq char
				4. Sort them and replace in '?'

Find Ugly number which is divisible only by 2, 3 and 5	https://leetcode.com/problems/ugly-number-ii/description/?envType=daily-question&envId=2024-08-18	1. Using DP, initiate dp[i]=1, 2. 3 variables two, three and five as 0 3. For 1 to 1000 4. Find the Min(2*dp[two], 3* dp[three], 5* dp[five]) 5. Which ever is small increment the value		Logic is only the values already present in the dp[] multiple will have exact multiple.. 1,2,3,4,5,6,8,9 etc., Other values like other prime numbers will be automatically removed

My Calendar booking	https://leetcode.com/problems/my-calendar-i/?envType=daily-	Integer floorKey = calendar.floorKey(start);
		if (floorKey !=
				1. FloorKey of treemap gives the key which has less than or equal

	envType=daily-question&envId=2024-09-26	null && calendar.get(floorKey) > start) return false;
		Integer ceilingKey = calendar.ceilingKey(start);
		if (ceilingKey != null && ceilingKey < end) return false;
				has less than or equal to given number
				2. Ceiling key of treemap gives the key which has greater than or equalto given number

Divide intervals divide them into minimum groups	https://leetcode.com/problems/divide-intervals-into-minimum-number-of-groups/?envType=daily-question&envId=2024-10-12	public int minGroups(int[][] intervals) {
		int[] count = new int[1000002];
		for(int i =0; i < intervals.length; i++) {
		count[intervals[i][0]]++;
		count[intervals[i][1] + 1]--;
		}
		int max = 0;
		int currSum =0;
		for(int i =1; i < count.length; i++) {
		currSum += count[i];
		max = Math.max(currSum, max);
		}
		return max;
		}
				For solving in O(N) we can keep an array from 0 to max.. For every start do +1 and next element of end do -1.. So when we sum we can find overlaps