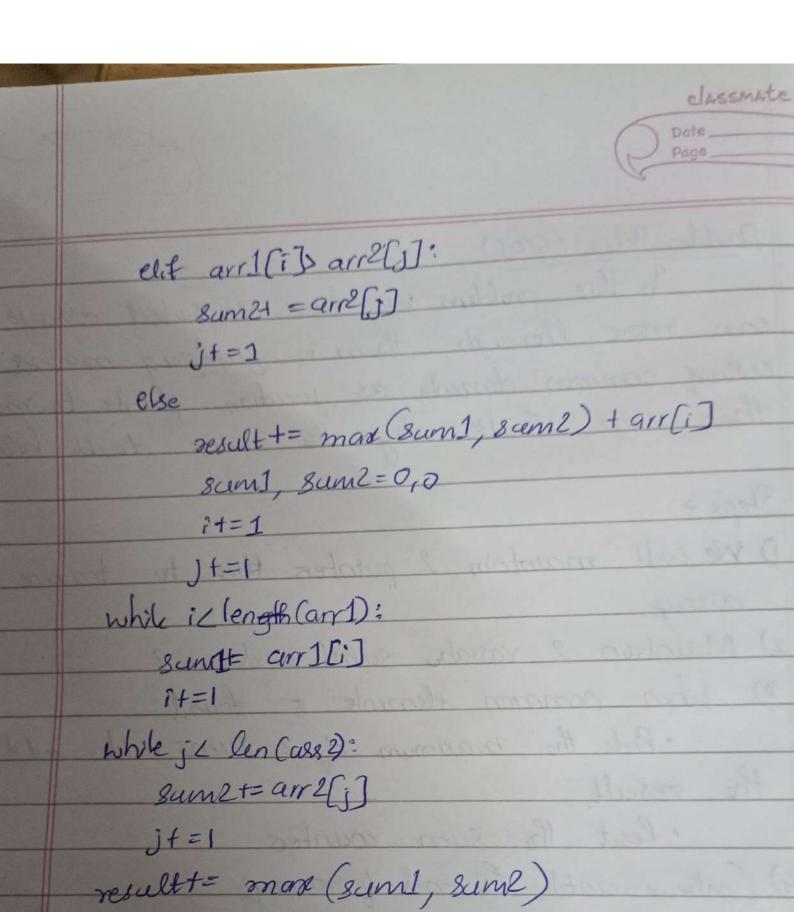
Darble Helix (SPOJ)
In this problem we have 2 sorted arrays and can move through them in zig-zag marries using common elements as junction points to maxing the sum. That's why the name double held. 1) We will maintain 2 pointers that to traverse buts 2) Maintain 2 variable sum1 and sum? 3) When common elements to found: · Prok the maximum of 2 sums and add to the result · Reset the sum counters 4) Continue certil the end of arrays let maxSumPath (asr1, ass2): ij = 0,0 Sum1, sum2= 0,0 Code? result=0 white illen (arr 1) and 54 lin (arr2): A ass[i] Carr2[j]: sum1+= arr[i] i+=1



return result

arr1 = [3,5,7,9,20,25,30,40,55,56,57,60,62]arr2 = [1,4,7,11,14,25,44,47,55,57,100]

Step	i (arr1)	j (arr2)	sum1	sum2	Common?	Result
1	3	1	3	1	No	0
2	5	4	8	5	No	0
3	7	7	15	12	Yes	15 (max(15,12) + 7)
4	9	11	9	11	No	15
5	20	14	29	25	No	15
6	25	25	54	50	Yes	54 (max(54,50) + 25)
7	30	44	30	44	No	79
8	40	47	70	91	No	79
9	55	55	125	193	Yes	272 (max(125,193) + 55)
10	56	57	56	57	No	327
11	57	57	113	114	Yes	384 (max(113,114) + 57)
12	60	100	60	100	No	384
13	62	End	122	100	No	450 (max(122,100))

Time complexity

Best (ase (O(N+M))

If both sequence have no common clineds,

we just trovere both away once I take the sume

of the larger one

* No switching between pashs

Waxt (ase (O(N+M))

* If there are many common clouds, we just)

keep resulting sum1 and sum2 at every common

mumber

* Still, since we only s(an each list one

the time complexity remains O(N+M)