Find a Pair With Given Difference

Tuesday, 9 November 2021 9:59 PM

pair having sum = target → two pointer
lo = 0
hi = n-1

array → 5 2 3 80 5 20

diff = 78

Logic for difference →

(17)

Sort →

Searching and Sorting

→ Sort

→ Binary Search.

* Time and Space
* Arrays.
* Two pointer Approach.

options: ① lo = 0, hi = n-1

Impact on difference →

Possibility ①

lo ++

Initially it was D_1

$D_1 - \Delta D_1 \Rightarrow D_2$

where $D_2 < D_1$

Possibility ②

hi --

Initially it was D_1

$D_1 - \Delta D_2 \Rightarrow D_2$

where $D_2 < D_1$

options ② lo = 0, hi = 1

Possibility ①

lo ++

Initially diff D_1

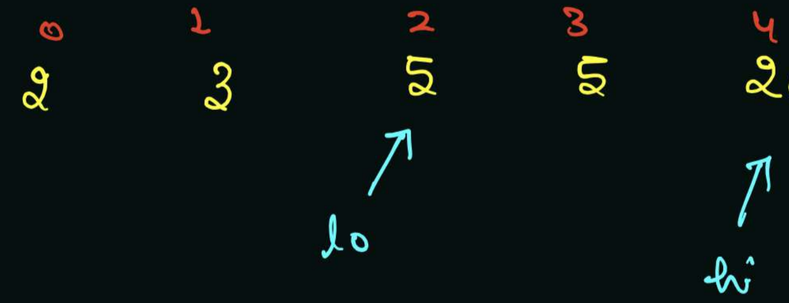
with lo++, difference decrease.

Possibility ②

hi ++

Initially it was D_1

with hi++, difference increase.



target → 15

$2n + n\log n$
traversal sorting

$O(n \log n)$

lo = ~~0~~ ~~1~~ 2

hi = ~~4~~ ~~3~~ ~~2~~ ~~1~~ 4

diff = ~~1~~ ~~2~~ ~~3~~ ~~18~~ ~~17~~ (15)

[5 + 20] =

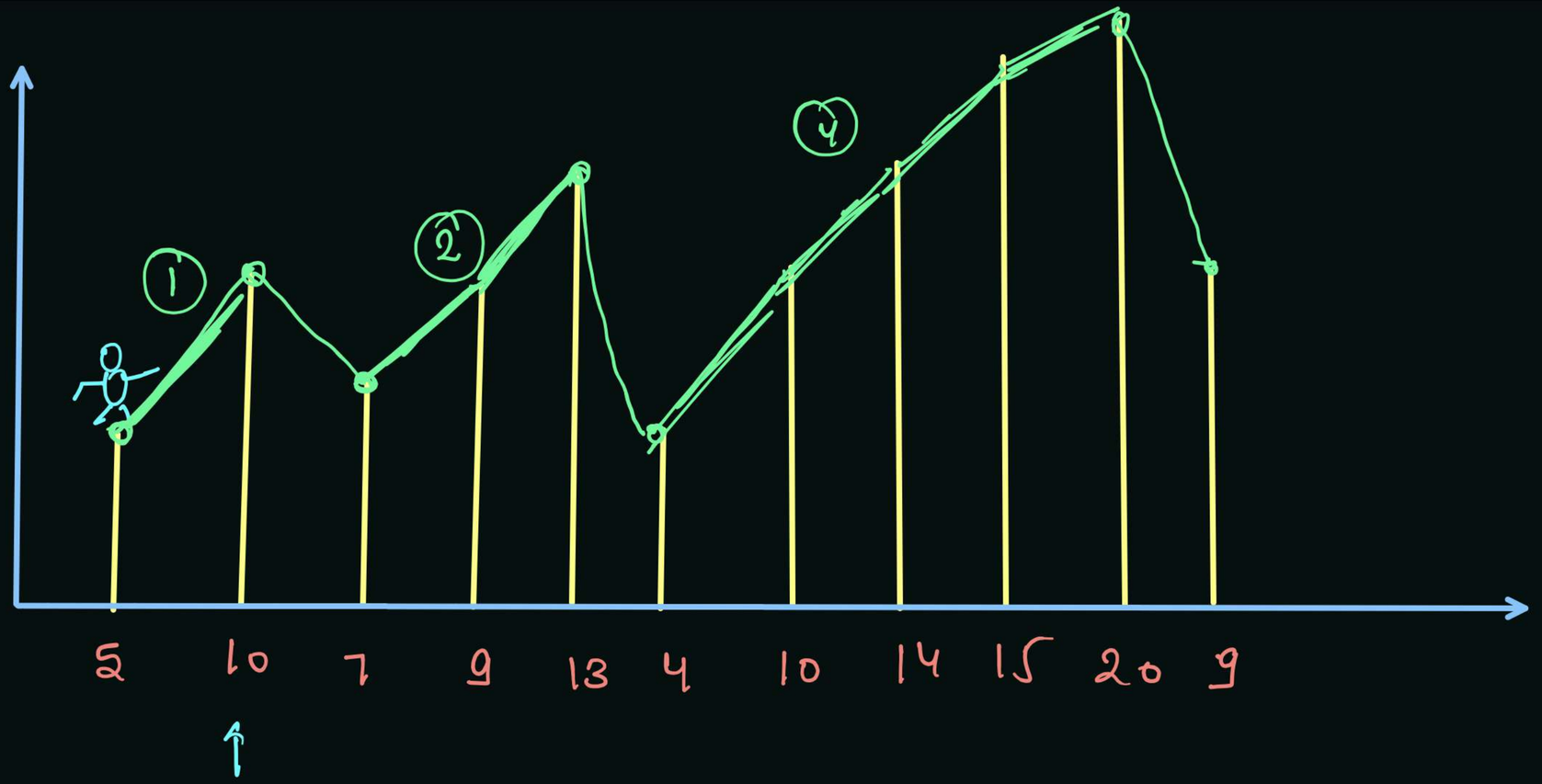
=

```
while (lo < hi) {
    difference = arr[hi] - arr[lo]

    if ( difference == target ) {
        pair = pair(arr[lo] + arr[hi]);
        return;
    } else if (difference > target){
        lo++;
    } else {
        hi++;
    }
}
```

Roof Top

Tuesday, 9 November 2021 9:59 PM




```
if( arr[i-1]  < arr[i] ) {
      count++;
}else {
      res = math.max(res, count);
      count = 0;
}
```

(4) Result

Maximize Sum Of Arr[i]*i Of and Array

Tuesday, 9 November 2021 10:00 PM

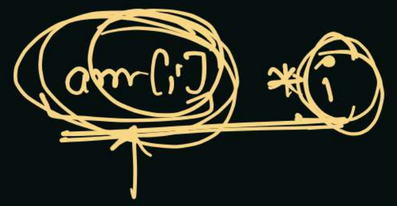
Maximum possible arrangements of arr's elements = n!

array → | 0 | 1 | 2 | 3 |
| a | b | c | a |

4! = 4 x 3 x 2 x 1 = 24

from all possible arrangements find max. sum of

$\sum_{i=0}^{n-1} arr[i] * i$



max. val will product with max. possible index

⇒ sum will be max.

i →	0	1	2
a	b	c
b	a	c
b	c	a
c	a	b
c	b	a
a	c	b

3! = 3 x 2 = 6

a * 0 + b * 1 + c * 2 = S_1

b * 0 + a * 1 + c * 2 = S_2

⋮

→ maximum Sum

→ Sort the array → $\sum_{i=0}^{n-1} arr[i] * i$

Maximum Sum in the configuration

Tuesday, 9 November 2021 10:34 PM

$\sum_{i=0}^{n-1} arr[i]*i$] Result] maximise.

0	1	2	3	
8	1	2	3	S_1
3	8	1	2	S_2
2	3	8	1	S_3
1	2	3	8	S_4

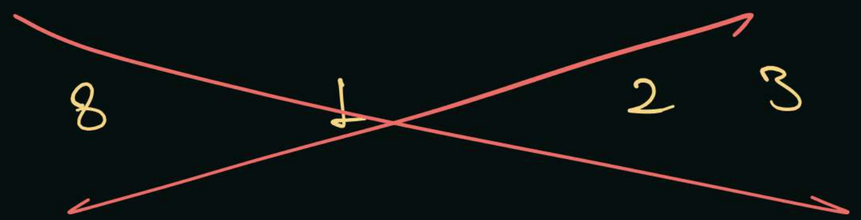
S_1 $8*0 + 1*1 + 2*2 + 3*3$

S_2 $S_1+8+1+2$ → $3*0 + 8*1 + 1*2 + 2*3$

→ max ?? → $3*3$

$2*0 + 3*1 + 8*2 + 1*3$

$1*0 + 2*1 + 3*2 + 8*3$



	0	1	2	3	4
array →	a	b	c	d	e

Sum of array $S = a+b+c+d+e$

~~S_0~~ $= a*0 + b*1 + c*2 + d*3 + e*4$

	0	1	2	3	4
R_0 (S_0) →	a	b	c	d	e

$S_1 = e*0 + a*1 + b*2 + c*3 + d*4$

$S_1 = S_0 + a+b+c+d+e - e - e*4$

$$\boxed{S_1 = S_0 + Sum - e*5} \quad \text{——— (1)}$$

	0	1	2	3	4
R_1 (S_1) →	e	a	b	c	d

$S_2 = d*0 + e*1 + a*2 + b*3 + c*4$

$S_2 = S_1 + e + a + b + c + d - d - d*4$

$$\boxed{S_2 = S_1 + S - d*5} \quad \text{——— (2)}$$

	0	1	2	3	4
R_2 (S_2) →	d	e	a	b	c

$S_3 = c*0 + d*1 + e*2 + a*3 + b*4$

$$\boxed{S_3 = S_2 + S - c*5} \quad \text{——— (3)}$$

	0	1	2	3	4
R_3 (S_3) →	c	d	e	a	b

	0	1	2	3	4
R_4 (S_4) →	b	c	d	e	a

general form $\boxed{S_i = S_{i-1} + S - n * arr[n-i]} \Rightarrow$

Steps to write code ⟶

① find sum of array and S_0

② Travel from i=1 and calculate S_i

$$S_i = S_{i-1} + Sum - n * arr[n-i]$$

③ Maximise S_i and S_{i+1} and hold res in a variable,

④ Return res.