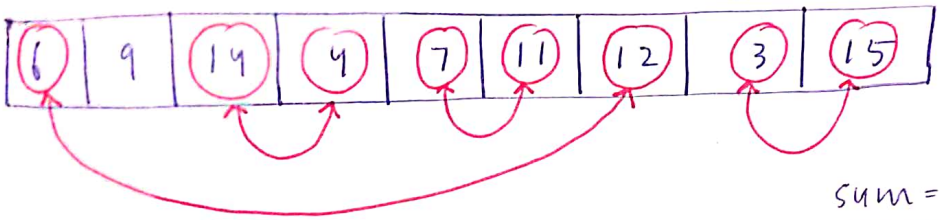
TARGET SUM PAIR

Hume n(logn) time complexity me solve krna hai ye Question!
Hume ek target milega Aur hume ek array bhi given hoga,
Hume vo sab pair of 2 element return krne hongy jo
uss target ke equal sum banayengy!



Given: 18

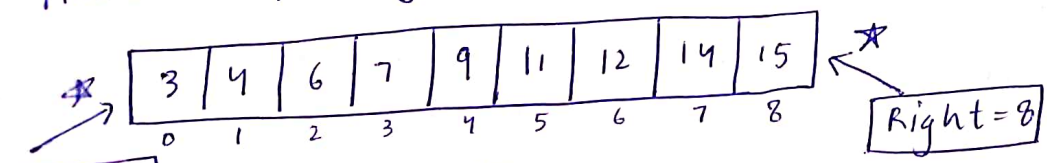
sum = 18 ∴ 6-12, 14-4, 7-11, 3-15 — 4 pair are present

Hume sabse phele array ko SORT krna hoga!

Arrays.sort() Function use krke humare array ko sort krlengy in n(logn) time complexity.

∴ 3 4 6 7 9 11 12 14 15 ∴ SORTED ARRAY

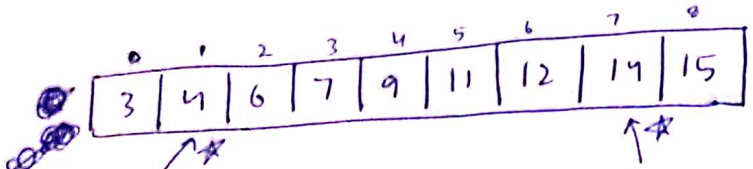
Aab sort krne ke baad [Left to Right] strategy use krengy! Aur iss strategy se pair nikalne me O(n) Time complexity lagegi!



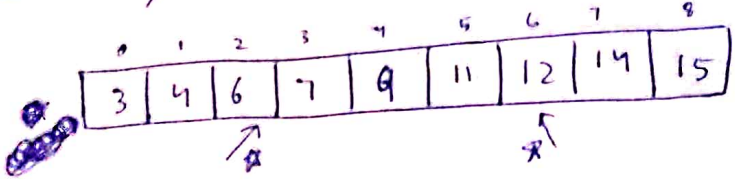
(3, 15)

sum = 3 + 15

if (sum > target) → Increase left Index
else if (sum < target) → Decrease right Index.
if (sum == target) → print left index and Right index elements and increase left and decrease right index.



(4, 14)



(6, 12)

3	4	6	7	9	11	12	14	15

(7, 11)

```
public static void main (String [] args){
 Scanner s = new Scanner (System.in);
 int target = s.nextInt();
 int n = s.nextInt();
 int [] arr = new int [n];
 for(int i = 0; i < arr.length; i++)
  {
    arr[i] = s.nextInt();
  }
 Arrays.sort(arr);  ]→ n log n (T.C)
 int left = 0;
 int right = arr.length - 1;
 while( left < right)
 { if(arr[left] + arr [right] > tar)
     {
       right --;
     }
    else if (arr[left] + arr [right] < tar)
     {
       left++;
     }
    else {
     System.out.println (arr[left]+ " "+ arr[right]);
        left++;
        right --;
   } }
  }
}
```