children & rides.

In an amusement park, there are rides being constructed for **N** children. Each ride has a maximum capacity limit of **B** for the total weight of the children on board.

You are given an array of weights of children, where **A[i]** is the weight of the **i-th** child, and an infinite number of rides. Each ride can carry **at most two** children at the same time, provided the sum of their weights is at most the ride's capacity limit.

Find the minimum number of rides to accommodate all the children.

[3, 2, 2, 0, 1] , B = 4

[3, 2, 2, 0, 1]

↓

[0, 1, 2, 2, 3]

(0, 3) → 1

(1, 2) → 2

(2) → 3.

[0, 1, 2, 2, 3] B = 4

[1, 1, 2, 2, 3, 4] B = 4

$n = 10^5$

```
count = 0;
Arrays.sort(arr);
l = 0;
r = arr.length - 1;
while (l < r) {
    sum = arr[l] + arr[r];
    if (sum <= B) {
        count++;
        l++;
        r--;
    }
    else {
        count++;
        r--;
    }
}
if (l == r) {
    count++;
}
```

Max Product of 3 Elements

Given an array of integers **A** of size **N**, return the maximum product of any three numbers from the array.

-1, 2, 1, 5 → sort

+ve

3 elements.

-ve +ve

2 neg

1 +ve.

-ve

-5 -4 -3 -2 -1

-6

first two & last one

-20.

arrays.sort(arr);

return max(arr[0]*arr[1]*arr[n-1], arr[n-1]*arr[n-2]*arr[n-3])

Search for a Range.

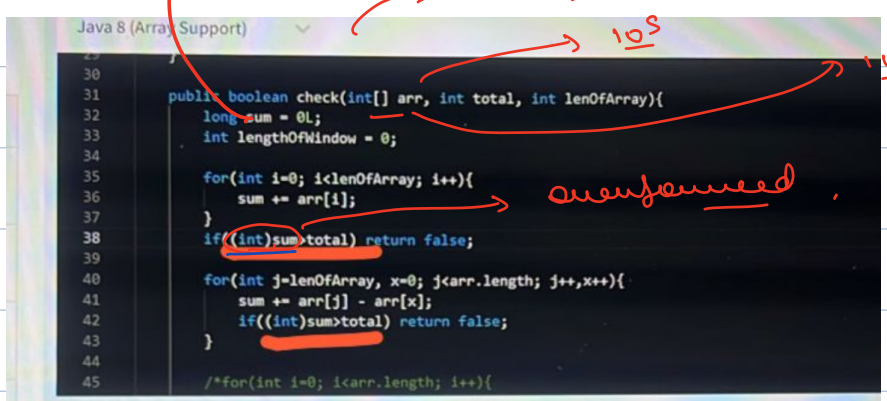
Given a **sorted array** of integers A (**0-indexed**) of size N, find the **left most** and the **right most** index of a given integer **B** in the array A.

Return an array of size 2, such that
First element = Left most index of B in A
Second element = Right most index of B in A.
If B is not found in A, return [-1, -1].

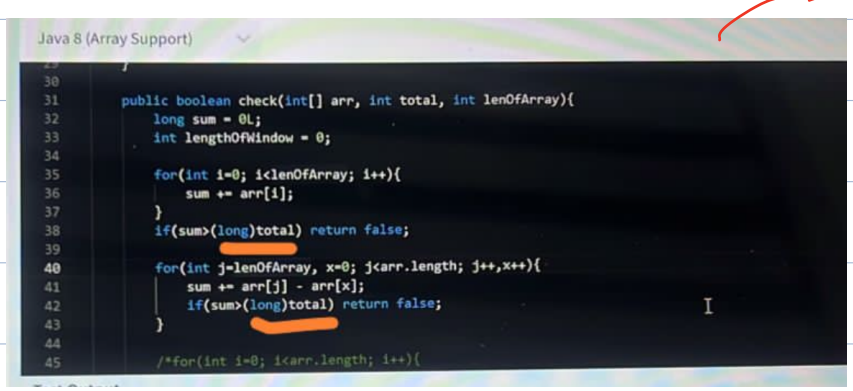
[5, 7, 7, 8, 8, 10] B = 8.

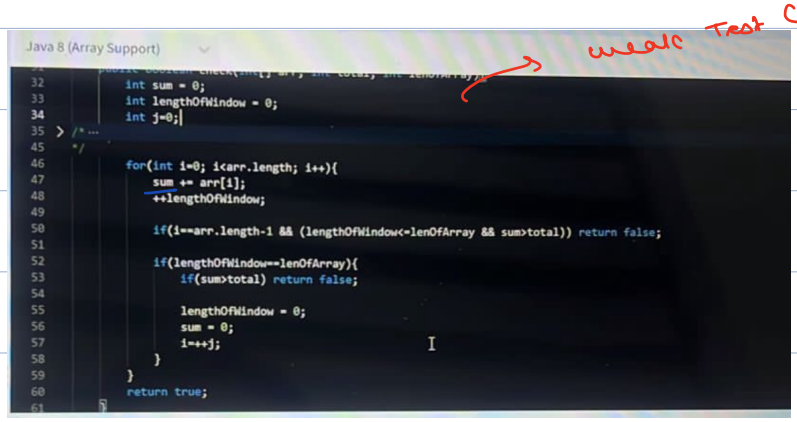
int sum = 10^4

Incorrect.



correct




Java 8 (Array Support)
int sum = 0;
int lengthOfWindow = 0;
int j=0;
/* ...
*/
for(int i=0; i<arr.length; i++){
sum += arr[i];
++lengthOfWindow;
if(i==arr.length-1 && (lengthOfWindow<=lenOfArray && sum>total)) return false;
if(lengthOfWindow==lenOfArray){
if(sum>total) return false;
lengthOfWindow = 0;
sum = 0;
i=++j;
}
}
return true;