Brute force

pair → first element of the possible

2,9
2,13
2,6

target = 16

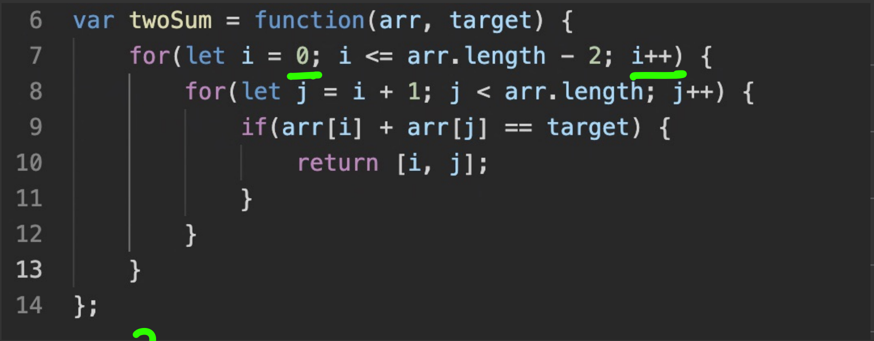
$[2, 9, 13, 6, 8, \overset{i}{7}, \underset{j}{5}]$

How about we try to form all possible pairs.

let's fix one element a[i] as the first element of the pair & then remaining element from index i+1 can become the 2nd element.

$$\boxed{a_i + a_j == target}$$

In one sec your computers can approx execute 10^8 instructions. TLE $n \to 10^4$



```
var twoSum = function(arr, target) {
    for(let i = 0; i <= arr.length - 2; i++) {
        for(let j = i + 1; j < arr.length; j++) {
            if(arr[i] + arr[j] == target) {
                return [i, j];
            }
        }
    }
};
```

$$\underbrace{n-1 + n-2 + n-3 \cdots\cdots +3+2+1}$$

$$\hookrightarrow \frac{n(n-1)}{2} \approx n^2$$

$i=0 \longrightarrow j \to [1, n-1] \longrightarrow n-1$ operation

$i=1 \longrightarrow j \to [2, n-1] \longrightarrow n-2$ operation

$i=2 \longrightarrow j \to [3, n-1] \longrightarrow n-3$ operation

$\vdots$

$i=n-2 \longrightarrow j \to [n-2, n-1] \longrightarrow 1$ operation

Let's consider that we had on array which was already arranged in asc order.

Can we do it in a single loop →

0 1 2 3 4 5 6

[2, 9, 13, 6, 8, 7, 5]

0 1 2 3 4 5 6

[2, 5, 6, 7, 8, 9, 13] ↗

0 1 2 3 4 5 6

S e

pair

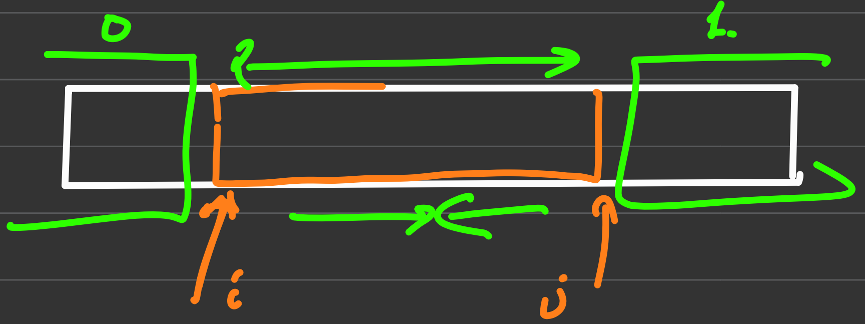
target = 13

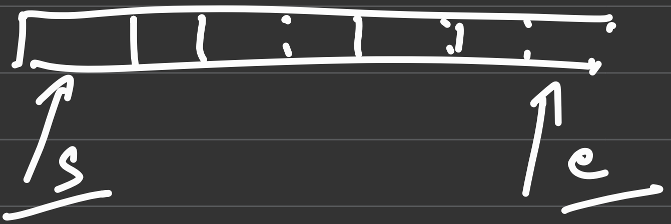
if question was asking just elements

two pointer technique

→ indexes

→ Sorting → 0, 1





$x < y$

x, y

y, x

comparator

How to make array arranged in inc order

→ function

arr.sort(fn);

arrange array in a particular order.

fn (x, y) {

it determines that when arr will arrange the elements whether x will come before y or not

}

both arguments are no.

```
arr.sort(fn);
```

```
fn(x, y){
    return x - y;
}
```

$x < y$

$x - y \rightarrow$ -ve x, y

$x > y$

$x - y \rightarrow$ +ve y, x

$y - x$

$x < y \rightarrow$ +ve y, x

$x > y \rightarrow$ x, y

lexicographical

arr.Sort( ); → Dictionary order

0 → A
1 → B
2 → C
3 → D
4 → E

[1, 10, 100, 2, 4, 3, 30]

[B, BA, BAA, C, E, D, DA]

B → BA → BAA → C → D → DA → E

[2, 9, 13, 6, 8, 7, 5]

fn (x, y) {

....

}

→ $\underline{-ve}$ x, y

→ $+ve$ y, x

$x < y$ → $(x - y)$ → $-ve$

$x > y$ → $x - y$ → $+ve$

a[i] → [el, ori i] [2, 9, 13, 6, 8, 7, 5]

array inside an array

9

[[2,0], [9,1], [13,2], [6,3], [8,4], [7,5], [5,6]]

[[2,0], [5,6], [6,3], [7,5], [8,4], [9,1], [13,2]]

s ↑S e e

arr.sort(fn),

fn (x, y) { x → [x1, x2] y [y1, y2]

x[0] − y[0] ×

}

$\underline{Q^n}$ Given an array arranged in ascendy order. And a target value. find the total number of pairs that sum upto the target.

Ex → [1, 2, 2, 3, 3, 4, 4, 4, 5, 5, 6, 6, 6],

Target → 8   ans → $\underline{13}$

$\underline{n \leq 10^5}$

(3+3+2+2+3)

target
↓
8

0	1	2	3	4	5	6	7	8	9	10	11	12
1	2	2	3	3	4	4	4	5	5	6	6	6

[1, 2, 2, 3, 3, 4, 4, 4, 5, 5, 6, 6, 6]

↑ s ↑ e

int a_s = 3

↓ we want freq of a_s

int a_e = 5

freq_of_s = ~~0~~ ~~1~~ 2
↓
x

freq_of_e = ~~0~~ ~~1~~ 2
↓
y

The moment we find a[s] + a[e] == target and a[s] != a[e]

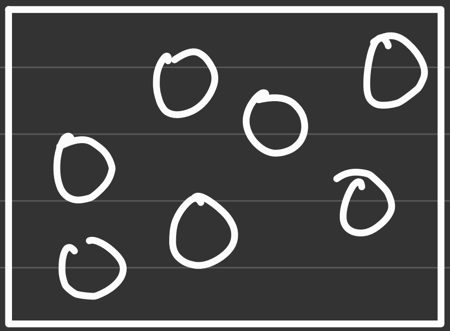
→ freq of a[s] → x → 2

→ freq of a[e] → y → 3

count += x*y

for → a[s] == a[e] → (e - s + 1) → $^{(e-s+1)}C_2 \rightarrow \frac{2\times(2-1)}{2}$

2 ←



$$\underline{\underline{{}^{x}C_2}}$$

$$\frac{x!}{2!\,(x-2)!} \rightarrow \frac{x \times (x-1)}{2}$$

for a[s] == a[e]

$$z = \frac{(e-s+1)}{}$$

$$\text{count} += \frac{(z) \times (z-1)}{2}$$