Two Sum

inp: 2 7 11 15 , target = 9
opt: 0, 1

→ Brute force - $O(n^2)$

Pseudo code

```
ans.
for i:0 → n-1
    for j:i+1 → n-1
        if arr[i] + arr[j] = target
            ans.push(i)
            ans.push(j)
            break;
return ans;
```

→ Second Approach

in this approach "Pair" vector that store element and indices

and sort the element.

Pseudo code

```
vector<pair<int,int>>v;
for i:0 → n-1
    v.push_back({arr[i], i});
sort(v.begin(), v.end())
int i=0, j = arr.size()-1
while(i<j)
{   if(v[i].first + v[j].first == target)
        return {v[i].second, v[j].second};
    else if(v[i].first + v[j].first > target)
        j--
    else
        i++
}
return {};
```

Dry run

arr: 2, 7, 11, 15 target = 9

v

2	0
7	1
11	2
15	3

After sort

2	0
7	1
11	2
15	3

i	j	condition	operation
0	3	17 > 9	j--;
0	2	13 > 9	j--;
0	1	9 == 9	return;

third Approach

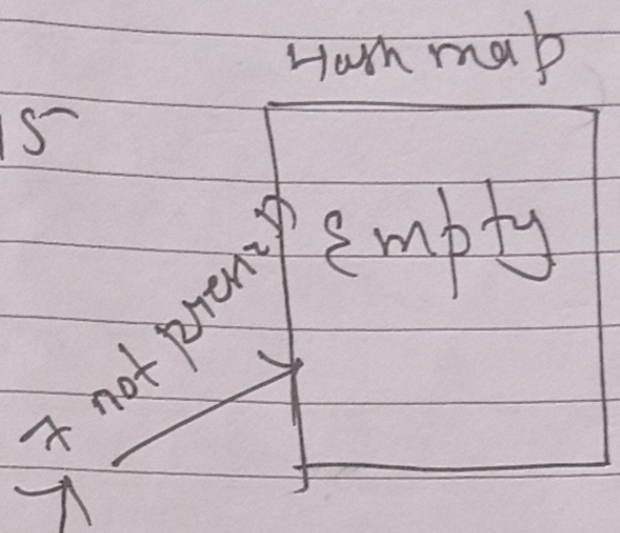
in this approach we create a hashmap and find complement of target and current element, then search complement in hash map.

Pseduco Pseudo code

```
unordered_map<int,int> m;
for i : 0 → n-1 {
    int c = target - arr[i];
    if (m.find(c) != m.end()){
        return {m[c], i};
    }
    m[arr[i]] = i;
}
```

Dry run.

inp: 2, 7, 11, 15
t = 9



C	condition	Operation
9 - 2 = 7	~~false~~	9 = 0
9 - 7 = 2	true	return (0, 1)

Hash map

2	0