2 SUM PROBLEM

☆ In this problem we are supposed to find two elements which add up to the target sum.

Brute force solution is to iterate through the entire array for each element and finding if they add up.

Better solution is to apply hashing using a hashmap. We iterate through the entire array and hash them one by one. While hashing them we simultaneously check for the difference's existence within the hashmap.

Pseudocode :

```
TwoSumProblem(arr, N, K) {
    map mpp
    for(int i = 0 → N) {
        a = arr[i]
        r = K - a
        if(mpp.find(r) != mpp.end()) {
            return {mpp[r], i}
        }
        mpp[a] = i
    }
}
```

Optimal solution is used if map is

not allowed. We sort the array and utilize the two pointer approach. One pointer starts at the start while the other starts at the end. We add both of them up, if sum is reached we return, else if sum is lesser than desired, we increment left pointer and if its more we decrement right pointer.

Pseudocode :

```
// Assuming array is already sorted
TwoSumProblem (arr , N , K) {
    i , j = 0 , N - 1
    while (j >= i) {
        if (arr [i] + arr [j] == K) {
            return {i , j}
        } else if (arr [i] + arr [j] > K) {
            j --
        } else {
            i ++
        }
    }
}
```