**Count All ((i,j) pairs such that b[i] - b[j] == k (count of such pairs.) [i<j]**

**Sample Testcase :**

**Arr = [1 , 2 ,3 , 4 , 5] k = 2**

**Output : 2**

**5 – 3 = 2**

**4 – 2 = 2**

**Arr = [ 3 , 5 , 7 , 9 , 6 , 1] k = 3**

**Output : 2**

**9 – 6 = 3**

**6 – 3 = 3**

Usually for 2 sum , we will find than **required = k – nums[j]** because **nums[i] + nums[j] = k** and we modify the eqn like that.

But here the eqn is **nums[i] – nums[j] = k** , so we need to make a simple change in the exisiting code tht **required = nums[j] + k.**

private static int TwoSumVariation(int [] nums , int k){

int n = nums.length();

//hash Map to store frequencies

for(int i = 0 ; i < n ; i++){

int required = nums[i] + k;

//Checking the existance of value

if(map.contains(required)){

cnt++;

}

//Update the index

map.put(nums[i] , i);

}

return cnt;

}