Valid Anagram

s1 = a b b a c → a-2, b-2, c-1

s2 = b a a c b → a-2, b-2, c-1

anagramic stoings : having same freq map.

s1 = a m p s a b m p → a-2, p-2, m-2, s-1, b-1

s2 = a a p p q s m → a-2, p-2, q-1, s-1, m-1

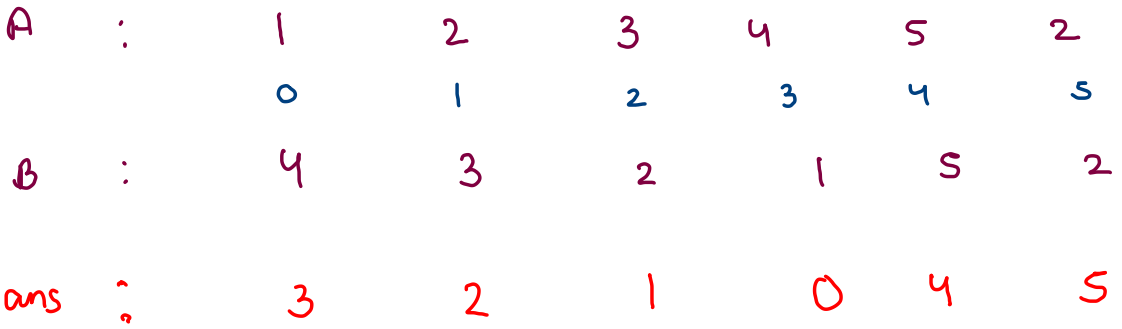
1) create fmap s1

2) cancel out fmap s2

Find Anagram Mappings

1. You are given two integer arrays(A and B), where B is an anagram of A.
2. B is an anagram of A means B is made by randomizing the order of the elements in A.
3. For every element in A, you have to print the index of that element in B.

```
6
1 2 3 4 5 2
4 3 2 1 5 2
```



A :	2	3	4	1	2	3	2
	0	1	2	3	4	5	6
B :	3	4	2	1	2	2	3
ans :	2	0	1	3	4	6	5

Front ran O(1)
-> LL
-> queue

Travel B, R -> L create a hm, key -> val

value -> AL < Int >

all occ of val in B

3 ->

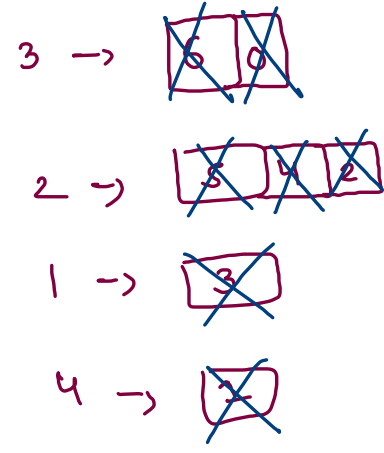
2 ->

1 ->

4 ->

```java
for(int i=B.length-1; i >= 0;i--) {
    int ele = B[i];

    if(map.containsKey(ele) == false) {
        ArrayList<Integer>list = new ArrayList<>();
        list.add(i);
        map.put(ele,list);
    }
    else {
        ArrayList<Integer>list = map.get(ele);
        list.add(i);
    }
}


int[]ans = new int[A.length];

for(int i=0; i < A.length;i++) {
    ArrayList<Integer>list = map.get(A[i]);
    int occ = list.remove(list.size()-1);

    ans[i] = occ;
}
```

A :	2	3	4	1	2	3	2
	0	1	2	3	4	5	6
B :	3	4	2	1	2	2	3
ans :	2	0	1	3	4	6	5



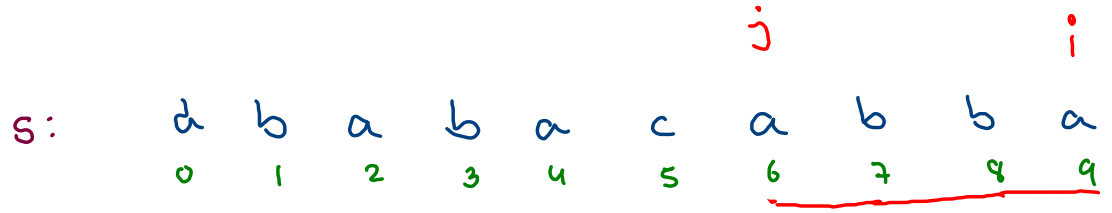
Find All Anagrams In A String

p = b a b a

ans → [0, 6]

are Maps Equal (pm, wm)

↳ O(26)



wm: a - 2
b - 2

pm: a - 2
b - 2

p: b a b a

ans: 1, 6

```
//freq map of first window in s
for(int i=0; i < p.length();i++) {
    char ch = s.charAt(i);
    int nf = wm.getOrDefault(ch,0) + 1;
    wm.put(ch,nf);
}


int j = 0;

for(; i < s.length();i++,j++) {
    if(areMapsEquals(wm,pm) == true) {
        ans.add(j);
    }

    //aquire ith char
    char chi = s.charAt(i);
    int nfi = wm.getOrDefault(chi,0) + 1;
    wm.put(chi,nfi);

    //release jth char
    char chj = s.charAt(j);

    if(wm.get(chj) == 1) {
        wm.remove(chj);
    }
    else {
        int nfj = wm.get(chj) - 1;
        wm.put(chj,nfj);
    }
}

if(areMapsEquals(wm,pm) == true) {
    ans.add(j);
}
```

s : d b a b a c a b b a

0 1 2 3 4 5 6 7 8 9

j i

p : b a b a

wm :

b - 2

a - 2

pm : a - 2

b - 2

ans : 1 , 6