String & character

↳ Practice Questions

Isogramic String (22 July)

Problem | Submissions | Leaderboard | Discussions

Given a string s in its lower-case, you have to decide if its an Isogram or not. A string is said to be an Isogram, if all the letters in the string, occur only once in it.

str = "machine" ; → Isogram

str = "qqwerqww"; → Not an Isogram

Sample Input 0

```
machine
```

Sample Output 0

```
Isogram
```

Sample Input 1

```
qqwerqww
```

Sample Output 1

```
Not an Isogram
```

m a c h i n e → 7

i = 0 < 7(7)

j = 1 2 3 4 < 7(7)

j = 2

m == a X

m == c X

m == h X

m == i X

m == n X

m == e X

a == c

c == h

i = 0

j = 1

(q == q)

A1

```
for (int i=0; i < str.length(); i++) {        → traversing on every character
    for (int j = i+1; j < str.length(); j++) {
        if( str.charAt(i) == str.charAt(j) )  return "Not an Isogram";
    }
}
return "Isogram";
```

```java
public static String checkIsogram(String str){
    for(int i=0;i<str.length();i++){
        for(int j=i+1;j<str.length();j++){
            if(str.charAt(i) == str.charAt(j)){
                return "Not an Isogram";
            }
        }
    }

    return "Isogram";
}
```

n

1, 2, 3, 4

1 + 2 + 3 + 4 + n

$\frac{n(n+1)}{2} = \frac{n^2+n}{2}$

$O(n^2)$

T.C = ? → $O(n^2)$

S.C = ? → O(1) or O(constant)

A2

26

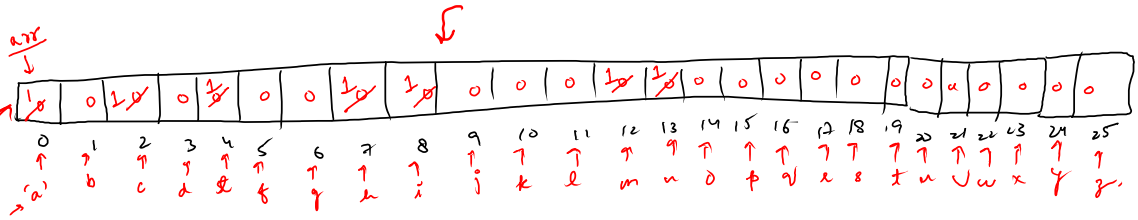
T.C. >

str = "machine";

⇒ 26 ⇒ smaller case.

→ O(n) + O(26)

→ T.C ⇒ O(n)



⇒ storing the frequency/count of every character present in my given string.

'm' - 'a' = 12

arr[12] += 1

'a' - 'a' = 0

'c' - 'a' = 2

'u' - 'a' = 7

'i' - 'a' = 8

'n' - 'a' = 13

'e' - 'a' = 4

(1) Making an array of size = 26

(2) Virtually mapping indexing with alphabets.

(3) Iterating over the given string str.

(4) Iterating over the arr.

↳ any index value > 1 → Not an Isogram. ✗

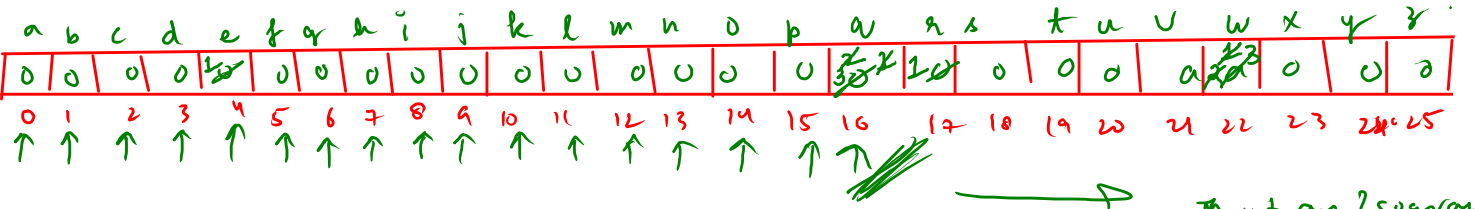
return "Isogram";

→ S.C → O(26)

⇓

nearly constant

qqwerqwww :



Not an Isogram

'q' - 'a' = 16

'q' - 'a' = 16

'w' - 'a' = 22

'e' - 'a' = 4

'r' - 'a' = 17

'w' - 'a' = 16

'w' - 'a' = 22

'w' - 'a' = 22

Approach-01

```java
public static String checkIsogram(String str){
    for(int i=0;i<str.length();i++){
        for(int j=i+1;j<str.length();j++){
            if(str.charAt(i) == str.charAt(j)){
                return "Not an Isogram";
            }
        }
    }

    return "Isogram";
}
```

5-6 min

T.C. ⇒ $O(n^2)$

S.C ⇒ $O(1)$

Approach-02

```java
public static String checkIsogram2(String str){
    int freq[] = new int[26];

    for(int i=0;i<str.length();i++){
        char ch = str.charAt(i);
        int index = ch-'a';
        freq[index]+=1;
    }

    for(int i=0;i<26;i++){
        if(freq[i]>1){
            return "Not an Isogram";
        }
    }

    return "Isogram";
}
```

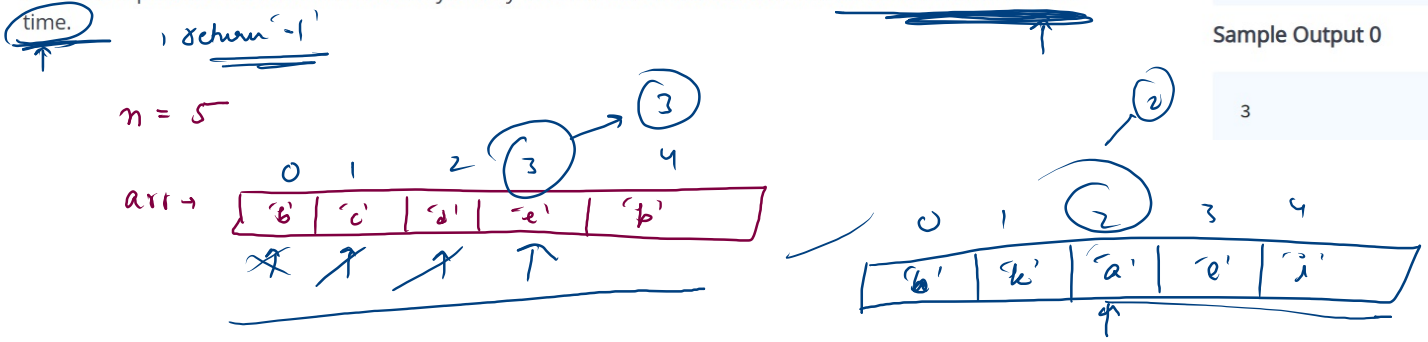
T.C. ⇒ $O(n)$

S.C ⇒ $O(26)$

Print First Vowel occurrence (22 july)

Problem | Submissions | Leaderboard | Discussions

Take n as an integer input. Declare the first array of size n that stores values of char data-type. Then take n character inputs and store them in the array one by one. Print the index at which the vowel occurs for the first time.



Sample Input 0

```
5
b
c
d
e
p
```

Sample Output 0

```
3
```

```java
public static boolean checkVowel(char ch){
    if(ch == 'a' || ch == 'e' || ch == 'i' || ch == 'o' || ch == 'u'){
        return true;
    }

    return false;
}

public static int firstVowelOccurence(char arr[], int n){
    for(int i=0;i<n;i++){
        if(checkVowel(arr[i]) == true){
            return i;
        }
    }
    return -1;
}

public static void main(String[] args) {
    /* Enter your code here. Read input from STDIN. Print output to STDOUT. Your class shoul
    Scanner scn = new Scanner(System.in);
    int n =scn.nextInt();
    char arr[] = new char[n];
    for(int i=0;i<n;i++){
        arr[i] = scn.next().charAt(0);
    }

    System.out.println(firstVowelOccurence(arr,n));
}
```

Check Anagram (22 July)

Problem | Submissions | Leaderboard | Discussions

Take two Strings as input and check whether they are anagram or not. Print True if they are anagram else print False.

Anagram: An anagram is a word or phrase formed by rearranging the letters of a different word or phrase, typically using all the original letters exactly once.

Eg: "peek" and "keep" are anagrams.

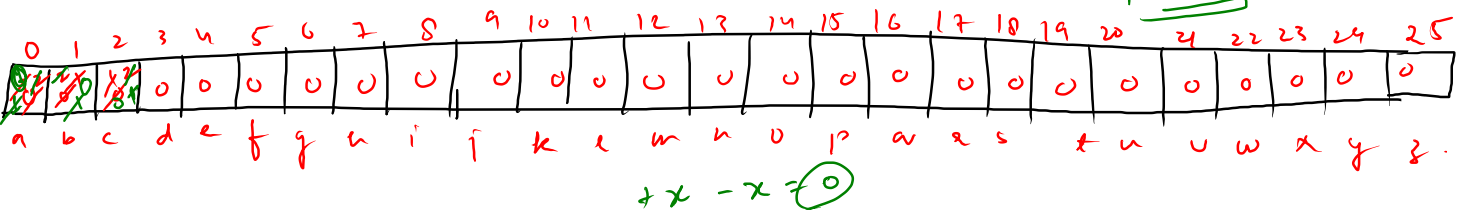
Note: All characters will be in lowercase.

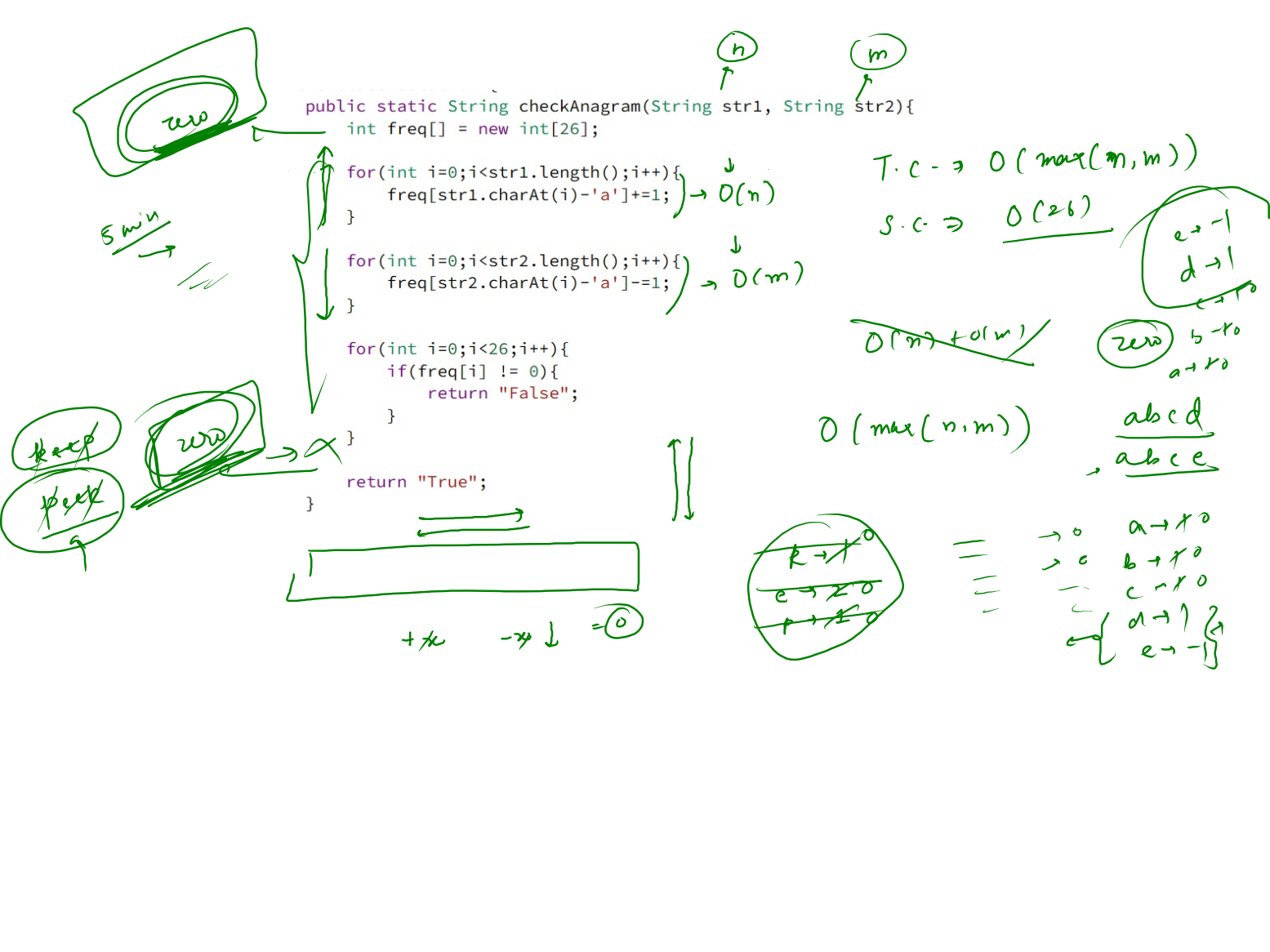
Sample Input 0

```
abccba
cbaabc
```

Sample Output 0

```
True
```




public static String checkAnagram(String str1, String str2){
int freq[] = new int[26];
for(int i=0;i<str1.length();i++){
freq[str1.charAt(i)-'a']+=1;
}
for(int i=0;i<str2.length();i++){
freq[str2.charAt(i)-'a']-=1;
}
for(int i=0;i<26;i++){
if(freq[i] != 0){
return "False";
}
}
return "True";
}
T.C -> O(max(n,m))
S.C -> O(26)
O(n)
O(m)
O(n)+O(m)
O(max(n,m))
abcd
abce

Is it a Pangram String? (22 July)

Problem | Submissions | Leaderboard | Discussions

Given a string str, you have to decide whether it is a Pangram or not.

Sample Input 0

```
the quick brown fox jumps over the lazy dog
```

Sample Output 0

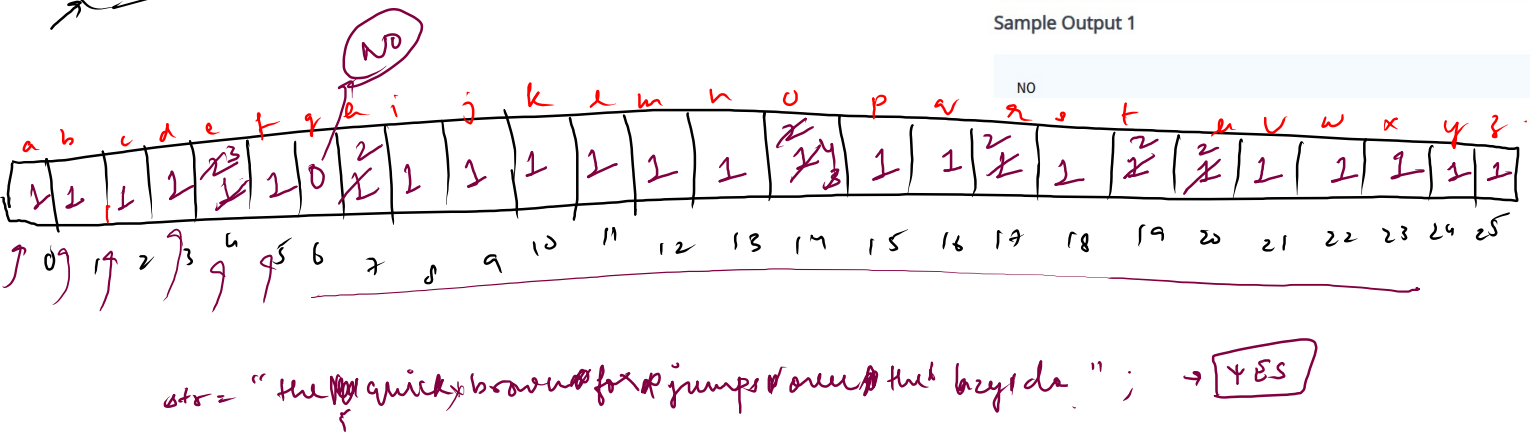
```
YES
```

Sample Input 1

```
abcdfeghijklpmnawrdfadfk
```

Sample Output 1

```
NO
```



str = "the quick brown fox jumps over the lazy dog" ; → YES

5 min

```java
public class Solution {
    public static String checkPangram(String str){
        int freq [] = new int[26];
        for(int i=0;i<str.length();i++){
            char ch = str.charAt(i);
            if(ch != ' '){
                freq[ch-'a']+=1;
            }
        }

        for(int i=0;i<26;i++){
            if(freq[i] == 0){
                return "NO";
            }
        }

        return "YES";
    }

    public static void main(String[] args) {
        /* Enter your code here. Read input from STDIN. Prin
        Scanner scn = new Scanner(System.in);
        String str = scn.nextLine();

        System.out.println(checkPangram(str));
    }
}
```

Toggle Case (23 July)

Problem | Submissions | Leaderboard | Discussions

1. You are given a string that contains only lowercase and uppercase alphabets.
2. You have to toggle the case of every character of the given string.

Sample Input 0

```
NaVgurUkUL
```

Sample Output 0

```
nAvGURuKul
```

str = "NaVgurUkUL"

resstr = "nAvGURuKul"

Lowercase

char (ch - 'a' + 'A')

Uppercase → ch = 'C'

ch - 'A'

'C' - 'A'

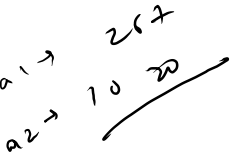
67 - 65 = 2 + 'a'

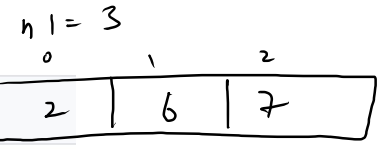
= 2 + 97 = (99) → 'c'

→ ch = ch - 'A' + 'a'

```java
public static String toggleCase(String str){
    String ans = "";
    for(int i=0;i<str.length();i++){
        char ch = str.charAt(i);
        if(ch>='A' && ch<='Z'){
          ans += (char)(ch-'A'+'a');
        }else{
            ans += (char)(ch-'a'+'A');
        }
    }

    return ans;
}

public static void main(String[] args) {
    /* Enter your code here. Read input from STDIN. Print outp
    Scanner scn = new Scanner(System.in);
    String str = scn.next();

    System.out.println(toggleCase(str));
}
```

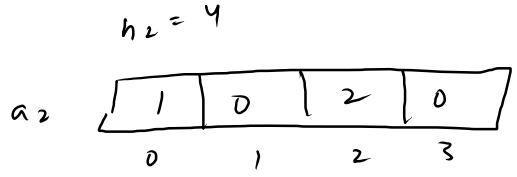
Difference of Arrays (2 July)



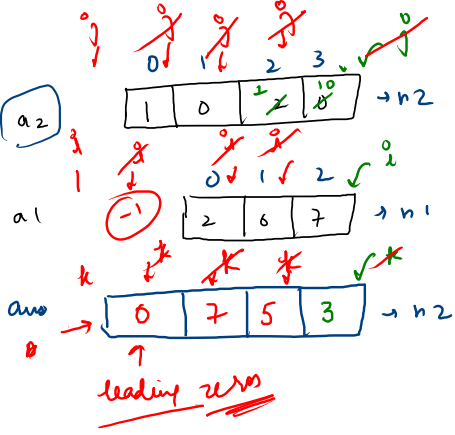


Problem | Submissions | Leaderboard | Discussions

1. You are given a number n1, representing the size of array a1.
2. You are given n1 numbers, representing elements of array a1.
3. You are given a number n2, representing the size of array a2.
4. You are given n2 numbers, representing elements of array a2.
5. The two arrays represent digits of two numbers.
6. You are required to find the difference of two numbers represented by two arrays and print the arrays a2 - a1
   - Assumption - number represented by a2 is greater.



a2 - a1



$j = n2-1$ while($k \geq 0$) borrow = ~~0~~ ~~1~~ Carry

$i = n1-1$ $a2[j] = 0 \; 1$

$k = n2-1$ * $a1[i] = 0$

diff = (0 - 1) - 2 = -3 + 10 = 7

borrow = 1

diff = (a2[j] - borrow) - a1[i]

-ve: diff += 10, borrow = 1

+ve: borrow = 0

→ borrow

diff = 0 - 0 - 7 = -7

diff = -7 + 10 = 3

borrow = 1

diff = (1 - 1) - 0 = 0

diff = (2 - 1) - 6 = 2 - 7 = -5

diff += 10 - 5 = 5

borrow = 1

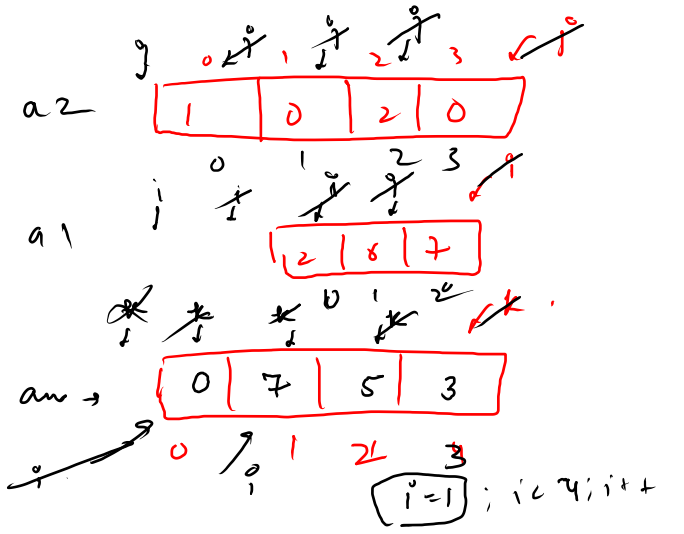
i - - - -

1 - - - -

k - - - -

```java
public static void differenceOfArrays(int a1[],int a2[],int n1,int n2){
    int ans[] = new int [n2];
    int i = n1-1;
    int j = n2-1;
    int k = n2-1;
    int borrow = 0;

    while(k>=0){
        int diff = a2[j]-borrow;

        if(i>=0){
            diff -= a1[i];
        }

        if(diff >=0){
            borrow=0;
        }else{
            diff += 10;
            borrow=1;
        }

        ans[k] = diff;
        i--;
        j--;
        k--;
    }

    i = 0;
    for(i=0;i<n2;i++){ // Remove leading zeros
        if(ans[i] !=0){
            break;
        }
    }

    for(;i<n2;i++){
        System.out.println(ans[i]);
    }
```

borrow = 0 → 1 → 1 → 1 → 0



diff = 0 - 0 - 7 + 10 = 3

= 2 - 1 = 1 - 6 = -5 + 10 = 5

= 0 - 1 = -1 - 2 = -3 + 10 = 7

= 1 - 1 = 0

7
5
3