Week 10:

ROLL NO.:240801160

Name: Keerthana G

Q1) Given a string, s, consisting of alphabets and digits, find the frequency of each digit in the given string.

Input Format

The first line contains a string, num which is the given number.

Constraints

$1 \leq len(num) \leq 1000$

All the elements of num are made of English alphabets and digits.

Output Format

Print ten space-separated integers in a single line denoting the frequency of each digit from 0 to 9.

Sample Input 0

a11472o5t6

Sample Output 0

0 2 1 0 1 1 1 1 0 0

Explanation 0

In the given string:

• 1 occurs two times.

• 2, 4, 5, 6 and 7 occur one time each.

• The remaining digits 0, 3, 8 and 9 don't occur at all.

Hint:

• Declare an array, freq of size 10 and initialize it with zeros, which will be used to count the frequencies of each of the digit occurring.

• Given a string, s, iterate through each of the character in the string. Check if the current character is a number or not.

• If the current character is a number, increase the frequency of that position in the freq array by 1.

• Once done with the iteration over the string, s, in a new line print all the 10 frequencies

starting from 0 to 9, separated by spaces.

Code:

```c
#include <stdio.h>
int main(){
    char str[1000];
    scanf("%s",str);
    int hash[10]={0,0,0,0,0,0,0,0,0,0};
    int temp;
    for(int i=0;str[i]!='\0';i++){
        temp=str[i]-'0';
        if(temp<=9&&temp>=0){
            hash[temp]++;
        }
    }
    for(int i=0;i<=9;i++){
        printf("%d ",hash[i]);
    }
    return 0;
}
```

OUTPUT:

	Input	Expected	Got	
✓	a11472o5t6	0 2 1 0 1 1 1 1 0 0	0 2 1 0 1 1 1 1 0 0	✓
✓	lw4n88j12n1	0 2 1 0 1 0 0 0 2 0	0 2 1 0 1 0 0 0 2 0	✓
✓	1v888861256338ar0ekk	1 1 1 2 0 1 2 0 5 0	1 1 1 2 0 1 2 0 5 0	✓

Passed all tests! ✓

Q2)

Today, Monk went for a walk in a garden. There are many trees in the garden and each tree has an English alphabet on it. While Monk was walking, he noticed that all trees with vowels on it are not in good state. He decided to take care of them. So, he asked you to tell him the count of such trees in the garden.

Note: The following letters are vowels: 'A', 'E', 'I', 'O', 'U', 'a', 'e', 'i', 'o' and 'u'.

Input Format:
The first line consists of an integer T denoting the number of test cases.
Each test case consists of only one string, each character of string denoting the alphabet (may be lowercase or uppercase) on a tree in the garden.

Output Format:
For each test case, print the count in a new line.

Constraints:
$1 \le T \le 10$
$1 \le$ length of string ≤ 105

Sample Input
2
nBBZLaosnm
JHkIsnZtTL

Sample Output
2
1

Explanation
In test case 1, a and o are the only vowels. So, count=2
Brief Description: Given a string S you have to count number of vowels in the string.

Solution 1:
For each vowel, count how many times it is appearing in the string S. Final answer will the sum of frequencies of all the vowels.

Solution 2:
Iterate overall all the characters in the string S and use a counter (variable) to keep track of number of vowels in the string S. While iterating over the characters, if we encounter a vowel, we will increase the counter by 1.

Time Complexity: O(N) where N is the length of the string S. Space Complexity: O(N)

Code:

```c
#include <stdio.h>
int main(){
    int t;
    scanf("%d",&t);
    while (t--){
        char str[100000];
        int count=0;
        scanf("%s",str);
        for (int i=0;str[i]!='\0';i++){
            char c=str[i];
            if ((c=='a')||(c=='e')||(c=='i')||(c=='o')||(c=='u')||(c=='A')||(c=='E')||(c=='I')||(c=='O')||(c=='U'))
                count++;
        }
        printf("%d\n",count);
    }
}
```

OUTPUT:

	Input	Expected	Got	
✓	2 nBBZLaosnm JHkIsnZtTL	2 1	2 1	✓
✓	2 nBBZLaosnm JHkIsnZtTL	2 1	2 1	✓

Passed all tests! ✓

Q3) Given a sentence, s, print each word of the sentence in a new line.

Input Format

The first and only line contains a sentence, s.

Constraints

$1 \le len(s) \le 1000$

Output Format

Print each word of the sentence in a new line.

Sample Input

This is C

Sample Output

This

is

C

Explanation

In the given string, there are three words ["This", "is", "C"]. We have to print each of these words in a new line.

Hint

Here, once you have taken the sentence as input, we need to iterate through the input, and keep printing each character one after the other unless you encounter a space. When a space is encountered, you know that a token is complete and space indicates the start of the next token after this. So, whenever there is a space, you need to move to a new line, so that you can start printing the next token.

Code:

```c
#include <stdio.h>
int main(){
    char s[1000];
    scanf("%[^\n]s",s);
    for(int i=0;s[i]!='\0';i++){
        if(s[i]!=' ')
        printf("%c", s[i]);
        else
        printf("\n");
    }
    return 0;
}
```

OUTPUT:

	Input	Expected	Got	
✓	This is C	This is C	This is C	✓
✓	Learning C is fun	Learning C is fun	Learning C is fun	✓

Passed all tests! ✓

Q4)

Input Format
You are given two strings, a and b, separated by a new line. Each string will consist of lower-case Latin characters ('a'-'z').

Output Format
In the first line print two space-separated integers, representing the length of a and b respectively.
In the second line print the string produced by concatenating a and b (a + b).
In the third line print two strings separated by a space, a' and b'. a' and b' are the same as a and b, respectively, except that their first characters are swapped.

Sample Input
abcd
ef

Sample Output
4 2
abcdef
ebcd af

Explanation
a = "abcd"
b = "ef"
|a| = 4
|b| = 2
a + b = "abcdef"
a' = "ebcd"
b' = "af"

Code:

```
1  #include <stdio.h>
2  int main(){
3      char str1[10],str2[10],t;
4      int i=0,j=0;
5      int count1=0,count2=0;
6      scanf("%s",str1);
7      scanf("%s",str2);
8      while (str1[i]!='\0')
9      {
10         count1++;
11         i++;
12     }
13     while(str2[j]!='\0')
14     {
15         count2++;
16         j++;
17     }
18     printf("%d %d\n",count1,count2);
19     printf("%s%s\n",str1,str2);
20     t=str1[0];
21     str1[0]=str2[0];
22     str2[0]=t;
23     printf("%s %s",str1,str2);
24     return 0;
25 }
```

OUTPUT:

	Input	Expected	Got	
✓	abcd ef	4 2 abcdef ebcd af	4 2 abcdef ebcd af	✓

Passed all tests! ✓