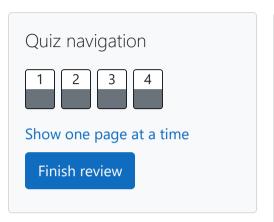
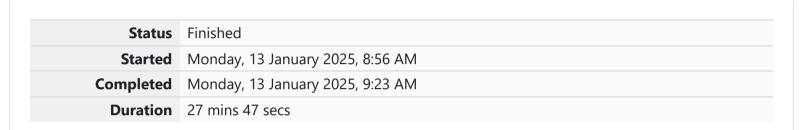
# GE23131-Programming Using C-2024





Question **1** 

Correct

Marked out of 1.00

▼ Flag question

Given a string,  $\mathbf{s}$ , 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**

## Sample Input 0

a11472o5t6

#### **Sample Output 0**

0210111100

## **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.

**Answer:** (penalty regime: 0 %)

```
16
17
18
for(int i=0;i<=9;i++)
19 v
{
20
printf("%d ",hash[i]);
21
22
}
23
return 0;
24
}
```

	Input	Expected Got		
~	a11472o5t6	0 2 1 0 1 1 1 1 0 0 0 2 1 0 1 1 1	100	~
~	lw4n88j12n1	0210100020 0210100	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! <

Question **2** 

Correct

Marked out of 1.00

Flag question

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:

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:

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

#### Constraints:

 $1 \le T \le 10$  $1 \le length of string \le 10^5$ 

#### **SAMPLE INPUT**

2

nBBZLaosnm

JHklsnZtTL

#### **SAMPLE OUTPUT**

2

1

#### **Explanation**

```
Answer: (penaity regime: 0 %)
```

```
#include<stdio.h>
         2
                               int main()
         3 ▼
                                                                int t;
                                                                scanf("%d",&t);
                                                                while(t--)
        7 🔻
                                                                                               char str[100000];
          9
                                                                                              int count=0;
                                                                                               scanf("%s",str);
 10
                                                                                               for(int i=0;str[i]!='\0';i++)
 11
12 🔻
 13
                                                                                                                              char c= str[i];
                                                                                                                              if((c=='a')||(c=='e')||(c=='i')||(c=='o')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=='u')||(c=
 14
 15
                                                                                                                              count++;
 16
                                                                                               printf("%d\n",count);
 17
 18
19
                                                                 return 0;
 20
```

	Input	Expected	Got	
~	2 nBBZLaosnm JHkIsnZtTL	2	2	<b>~</b>
~	2 nBBZLaosnm JHkIsnZtTL	2	2	~

Passed all tests! <

Marked out of 1.00 Flag question is C

#### **Input Format**

The first and only line contains a sentence, s.

#### **Constraints**

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

# **Output Format**

Print each word of the sentence in a new line.

# Sample Input 0

This is C

## **Sample Output 0**

This

## **Explanation 0**

```
Answer: (penalty regime: 0 %)
```

```
#include<stdio.h>
    int main()
 2
 3 ₹ {
        char s[1000];
        scanf("%[^\n]s",s);
 5
        for(int i=0;s[i]!='\0';i++)
 7 🔻
            if (s[i]!=' ')
            printf("%c",s[i]);
10
            else
            printf("\n");
11
12
13
        return 0;
14
15
```

	Input	Expected	Got	
<b>~</b>	This is C	This is C	This is C	~
~	Learning C is fun	Learning C is fun	Learning C is fun	<b>~</b>

Question **4** 

Correct

Marked out of 1.00

Flag question

#### **Input Format**

You are given two strings,  $\boldsymbol{a}$  and  $\boldsymbol{b}$ , 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  $\boldsymbol{a}$  and  $\boldsymbol{b}$  respectively.

In the second line print the string produced by concatenating  $\boldsymbol{a}$  and  $\boldsymbol{b}$  ( $\boldsymbol{a} + \boldsymbol{b}$ ).

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**

42

abcdef

ebcd af

```
b = "ef"
|a| = 4
|b| = 2
a + b = "abcdef"
a' = "ebcd"
b' = "af"
Answer: (penalty regime: 0 %)
      #include<stdio.h>
       int main()
    2
    3 ▼
           char str1[10],str2[10],t;
    4
           int i=0,j=0;
    5
           int count1=0, count2=0;
           scanf("%s",str1);
           scanf("%s",str2);
    8
           while(str1[i]!='\0')
    9
   10
   11
                count1++;
                i++;
   12
   13
   14
           while(str2[j]!='\0')
   15
   16 1
   17
                count2++;
   18
                j++;
   19
   20
   21
           printf("%d %d\n",count1,count2);
   22
           printf("%s%s\n",str1,str2);
   23
           t=str1[0];
   24
           str1[0]=str2[0];
   25
           str2[0]=t;
```

a = "abcd"

