Question **1**

Correct

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 ≤ len(num) ≤ 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.

Answer: (penalty regime: 0 %)

```c
#include <stdio.h>
#include <string.h>

int main() {
    char s[1001];
    int freq[10] = {0};     // Frequency array for digits 0-9

    scanf("%s", s);         // Read the input string

    // Traverse the string
    for(int i = 0; s[i] != '\0'; i++) {
        if(s[i] >= '0' && s[i] <= '9') {   // Check if character is a digit
            freq[s[i] - '0']++;            // Convert char digit to index
        }
    }

    // Print frequencies of digits 0 to 9
    for(int i = 0; i < 10; i++) {
        printf("%d", freq[i]);
        if(i < 9) printf(" ");
    }

    return 0;
}
```

	Input
✓	a11472o5t6
✓	lw4n88j12n1
✓	1v88886l256338ar0ekk
✓	8a36a4r26079184sr06u54rdr234u70653w7a32ol81o8980780958s3u5y27m69gr4a14382f8g552(
✓	b3n47b5xf13qlx233rg4u2c949i623e34nt5661se06b675utbpy258wz633855846l761d61x340h1v

Passed all tests! ✓

Question **2**

Correct

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 ≤ len(s) ≤ 1000

Output Format

Print each word of the sentence in a new line.

Sample Input 0

This is C

Sample Output 0

This

is

C

Explanation 0

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

Answer: (penalty regime: 0 %)

```c
#include <stdio.h>

int main() {
    char s[1001];

    // Read the full sentence including spaces
    fgets(s, sizeof(s), stdin);

    // Print each word on a new line
    for (int i = 0; s[i] != '\0'; i++) {
        if (s[i] == ' ') {
            printf("\n");   // New line when space is found
        } else {
            printf("%c", s[i]);
        }
    }

    return 0;
}
```

	Input	Expected	Got	
✓	This is C	This is C	This is C	✓
✓	Learning C is fun	Learning C is fun	Learning C is fun	✓
✓	How is that	How is that	How is that	✓
✓	The first and only line contains a sentence	The first and only line contains a sentence	The first and only line contains a sentence	✓

Passed all tests! ✓

Question **3**

Correct

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"

Answer: (penalty regime: 0 %)

```c
#include <stdio.h>
#include <string.h>

int main() {
    char a[1000], b[1000];

    // Read two strings
    scanf("%s", a);
    scanf("%s", b);

    // Print lengths
    printf("%lu %lu\n", strlen(a), strlen(b));

    // Print concatenation
    printf("%s%s\n", a, b);

    // Swap first characters (only if both strings are non-empty)
    char a_prime[1000], b_prime[1000];
    strcpy(a_prime, a);
    strcpy(b_prime, b);

    if (strlen(a) > 0 && strlen(b) > 0) {
        char temp = a_prime[0];
        a_prime[0] = b_prime[0];
        b_prime[0] = temp;
    }

    // Print swapped result
    printf("%s %s\n", a_prime, b_prime);

    return 0;
}
```

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

	Input	Expected	Got	
✓	abcd xy	4 2 abcdxy xbcd ay	4 2 abcdxy xbcd ay	✓

Passed all tests! ✓