
Started on Tuesday, 7 October 2025, 12:35 PM

State Finished

Completed on Thursday, 9 October 2025, 11:53 PM

Time taken 2 days 11 hours

Marks 10.00/10.00

Grade **100.00** out of 100.00

Question 1 | Correct Mark 1.00 out of 1.00

Objective:

Develop a Python program that takes an input string from the user and counts the number of occurrences of each vowel (a, e, i, o, u) in the string. The program should be case-insensitive, meaning it should treat uppercase and lowercase vowels as the same.

Description:

Vowels play a significant role in the English language and other alphabet-based languages. Counting vowels in a given string is a fundamental task that can be applied in various text processing applications, including speech recognition, linguistic research, and text analysis. The objective of this problem is to create a Python script that accurately counts and displays the number of times each vowel appears in a user-provided string.

Program Requirements:**Input:**

First line reading String as input, The string can contain any characters, including letters, numbers, and special characters.

Output:

Display the number of occurrences of each vowel in the string.

The output should list each vowel followed by its count.

Example:

Consider the following example for better understanding:

- **Input:** "Python Programming"
- **Output**

a = 1
e = 0
i = 1
o = 2
u = 0

For example:

Input	Result
Hello World	a = 0 e = 1 i = 0 o = 2 u = 0
Python	a = 0 e = 0 i = 0 o = 1 u = 0

Answer: (penalty regime: 0 %)

```
1 text=input()
2 text=text.lower()
3 vowels="aeiou"
4 for v in vowels:
5     print(v,"=",text.count(v))
```

	Input	Expected	Got	
✓	Hello World	a = 0 e = 1 i = 0 o = 2 u = 0	a = 0 e = 1 i = 0 o = 2 u = 0	✓
✓	AEIOU aeio u	a = 2 e = 2 i = 2 o = 2 u = 2	a = 2 e = 2 i = 2 o = 2 u = 2	✓
✓	Python	a = 0 e = 0 i = 0 o = 1 u = 0	a = 0 e = 0 i = 0 o = 1 u = 0	✓
✓	abcdefghijklmnopqrstuvwxyz	a = 1 e = 1 i = 1 o = 1 u = 1	a = 1 e = 1 i = 1 o = 1 u = 1	✓
✓	12345!@#\$\$AEIOU	a = 1 e = 1 i = 1 o = 1 u = 1	a = 1 e = 1 i = 1 o = 1 u = 1	✓

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

Question 2 | Correct Mark 1.00 out of 1.00

Given a number, convert it into corresponding alphabet.

Input	Output
1	A
26	Z
27	AA
676	YZ

Input Format

Input is an integer

Output Format

Print the alphabets

Constraints

$1 \leq \text{num} \leq 4294967295$

Sample Input 1

26

Sample Output 1

Z

For example:

Test	Result
<code>print(excelNumber(26))</code>	Z

Answer: (penalty regime: 0 %)

Reset answer

```
1 def excelNumber(n):
2     result=""
3     while n>0:
4         n-=1
5         result=chr(n%26+65)+result
6         n//=26
7     return result
```

	Test	Expected	Got	
✓	print(excelNumber(26))	Z	Z	✓
✓	print(excelNumber(27))	AA	AA	✓

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

Question 3 | Correct Mark 1.00 out of 1.00

Give a dictionary with value lists, sort the keys by summation of values in value list.

Input : test_dict = {'Gfg' : [6, 7, 4], 'best' : [7, 6, 5]}

Output : {'Gfg': 17, 'best': 18}

Explanation : Sorted by sum, and replaced.

Input : test_dict = {'Gfg' : [8,8], 'best' : [5,5]}

Output : {'best': 10, 'Gfg': 16}

Explanation : Sorted by sum, and replaced.

Sample Input:

2

Gfg 6 7 4

Best 7 6 5

Sample Output

Gfg 17

Best 18

For example:

Input	Result
2	Gfg 17
Gfg 6 7 4	Best 18
Best 7 6 5	

Answer: (penalty regime: 0 %)

```

1 n=int(input())
2 d={}
3 for _ in range(n):
4     parts=input().split()
5     key=parts[0]
6     values=list(map(int,parts[1:]))
7     d[key]=values
8 d_sum={k:sum(v) for k,v in d.items()}
9 for k,v in sorted(d_sum.items(),key=lambda x:x[1]):
10     print(k,v)

```

	Input	Expected	Got	
✓	2 Gfg 6 7 4 Best 7 6 5	Gfg 17 Best 18	Gfg 17 Best 18	✓
✓	2 Gfg 6 6 Best 5 5	Best 10 Gfg 12	Best 10 Gfg 12	✓

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

Question 4 | Correct Mark 1.00 out of 1.00

You are given a string **word**. A letter is called **special** if it appears both in lowercase and uppercase in **word**.

Your task is to return the number of **special** letters in **word**.

Constraints

- The input string **word** will contain only alphabetic characters (both lowercase and uppercase).
- The solution must utilize a dictionary to determine the number of special letters.
- The function should handle various edge cases, such as strings without any special letters, strings with only lowercase or uppercase letters, and mixed strings.

Examples

Example 1:

Input: word = "aaAbcBC"

Output: 3

Explanation:

The special characters in `word` are 'a', 'b', and 'c'.

Example 2:

Input: word = "abc"

Output: 0

Explanation:

No character in `word` appears in uppercase.

For example:

Test	Result
<code>print(count_special_letters("AaBbCcDdEe"))</code>	5

Answer: (penalty regime: 0 %)

[Reset answer](#)

```
1 def count_special_letters(word: str) -> int:
2     # Your implementation here
3     lower=set()
4     upper=set()
5     for ch in word:
6         if ch.islower():
7             lower.add(ch)
8         elif ch.isupper():
9             upper.add(ch.lower())
10    return len(lower & upper)
11    pass
```


	Test	Expected	Got	
✓	print(count_special_letters("AaBbCcDdEe"))	5	5	✓
✓	print(count_special_letters("ABCDE"))	0	0	✓
✓	print(count_special_letters("abcde"))	0	0	✓

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

Question 5 | Correct Mark 1.00 out of 1.00

Given an array of names of candidates in an election. A candidate name in the array represents a vote cast to the candidate. Print the name of candidates received Max vote. If there is tie, print a lexicographically smaller name.

Examples:

```
Input : votes[] = {"john", "johnny", "jackie",  
                  "johnny", "john", "jackie",  
                  "jamie", "jamie", "john",  
                  "johnny", "jamie", "johnny",  
                  "john"};
```

Output : John

We have four Candidates with name as 'John', 'Johnny', 'jamie', 'jackie'. The candidates John and Johnny get maximum votes. Since John is alphabetically smaller, we print it. Use dictionary to solve the above problem

Sample Input:

```
10  
John  
John  
Johnny  
Jamie  
Jamie  
Johnny  
Jack  
Johnny  
Johnny  
Jackie
```

Sample Output:

```
Johnny
```

Answer: (penalty regime: 0 %)

```

1 n=int(input())
2 votes=[input() for _ in range(n)]
3 count={}
4 for v in votes: count[v]=count.get(v,0)+1
5 print(sorted([k for k,c in count.items() if c==max(count.values())])[0])

```

	Input	Expected	Got	
✓	10 John John Johnny Jamie Jamie Johnny Jack Johnny Johnny Jackie	Johnny	Johnny	✓
✓	6 Ida Ida Ida Kiruba Kiruba Kiruba	Ida	Ida	✓

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

Question 6 | Correct Mark 1.00 out of 1.00

A sentence is a string of single-space separated words where each word consists only of lowercase letters. A word is uncommon if it appears exactly once in one of the sentences, and does not appear in the other sentence.

Given two sentences `s1` and `s2`, return a list of all the uncommon words. You may return the answer in any order.

Example 1:

Input: `s1 = "this apple is sweet", s2 = "this apple is sour"`

Output: `["sweet", "sour"]`

Example 2:

Input: `s1 = "apple apple", s2 = "banana"`

Output: `["banana"]`

Constraints:

`1 <= s1.length, s2.length <= 200`

`s1` and `s2` consist of lowercase English letters and spaces.

`s1` and `s2` do not have leading or trailing spaces.

All the words in `s1` and `s2` are separated by a single space.

Note:

Use dictionary to solve the problem

For example:

Input	Result
this apple is sweet this apple is sour	sweet sour

Answer: (penalty regime: 0 %)

```
1 s1=input()
2 s2=input()
3 d={}
4 for w in (s1+" "+s2).split():
5     d[w]=d.get(w,0)+1
6 print(" ".join([w for w,c in d.items() if c==1]))
```

	Input	Expected	Got	
✓	this apple is sweet this apple is sour	sweet sour	sweet sour	✓
✓	apple apple banana	banana	banana	✓

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

Question 7 | Correct Mark 1.00 out of 1.00

A sentence is a list of words that are separated by a single space with no leading or trailing spaces. Each word consists of lowercase and uppercase English letters.

A sentence can be shuffled by appending the 1-indexed word position to each word then rearranging the words in the sentence.

For example, the sentence "This is a sentence" can be shuffled as "sentence4 a3 is2 This1" or "is2 sentence4 This1 a3".

Given a shuffled sentence *s* containing no more than 9 words, reconstruct and return the original sentence.

Example 1:

Input:

is2 sentence4 This1 a3

Output:

This is a sentence

Explanation: Sort the words in *s* to their original positions "This1 is2 a3 sentence4", then remove the numbers.

Example 2:

Input:

Myself2 Me1 I4 and3

Output:

Me Myself and I

Explanation: Sort the words in *s* to their original positions "Me1 Myself2 and3 I4", then remove the numbers.

Constraints:

$2 \leq s.length \leq 200$

s consists of lowercase and uppercase English letters, spaces, and digits from 1 to 9.

The number of words in *s* is between 1 and 9.

The words in *s* are separated by a single space.

s contains no leading or trailing spaces.

Answer: (penalty regime: 0 %)

```
1 s=input()
2 words=s.split()
3 d={}
4 for w in words:
5     d[int(w[-1])]=w[:-1]
6 result=''
7 for i in range(1,len(words)+1):
8     result+=d[i] + ' '
9 print(result.strip())
```

	Input	Expected	Got	
✓	is2 sentence4 This1 a3	This is a sentence	This is a sentence	✓

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

//

Question 8 | Correct Mark 1.00 out of 1.00

In the game of Scrabble™, each letter has points associated with it. The total score of a word is the sum of the scores of its letters. More common letters are worth fewer points while less common letters are worth more points. The points associated with each letter are shown below:

Points Letters

1 A, E, I, L, N, O, R, S, T and U

2 D and G

3 B, C, M and P

4 F, H, V, W and Y

5 K

8 J and X

10 Q and Z

Write a program that computes and displays the Scrabble™ score for a word. Create a dictionary that maps from letters to point values. Then use the dictionary to compute the score.

A Scrabble™ board includes some squares that multiply the value of a letter or the value of an entire word. We will ignore these squares in this exercise.

[Sample Input](#)

REC

[Sample Output](#)

REC is worth 5 points.

For example:

Input	Result
REC	REC is worth 5 points.

Answer: (penalty regime: 0 %)

```

1 points={
2     1:"A E I L N O R S T U".split(),
3     2:"D G".split(),
4     3:"B C M P".split(),
5     4:"F H V W Y".split(),
6     5:"K".split(),
7     8:"J X".split(),
8     10:"Q Z".split()
9 }
10 score_dict={}
11 for score, letters in points.items():
12     for letter in letters:
13         score_dict[letter]=score
14 word=input().upper()
15 total=0
16 for ch in word:
17     total+=score_dict.get(ch,0)
18 print(f"{word} is worth {total} points.")

```


	Input	Expected	Got	
✓	GOD	GOD is worth 5 points.	GOD is worth 5 points.	✓
✓	REC	REC is worth 5 points.	REC is worth 5 points.	✓

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

Question 9 | Correct Mark 1.00 out of 1.00

Create a student dictionary for n students with the student name as key and their test mark assignment mark and lab mark as values. Do the following computations and display the result.

1. Identify the student with the highest average score
2. Identify the student who has the highest Assignment marks
3. Identify the student with the Lowest lab marks
4. Identify the student with the lowest average score

Note:

If more than one student has the same score display all the student names

Sample input:

4

James 67 89 56

Lalith 89 45 45

Ram 89 89 89

Sita 70 70 70

Sample Output:

Ram

James Ram

Lalith

Lalith

For example:

Input	Result
4	Ram
James 67 89 56	James Ram
Lalith 89 45 45	Lalith
Ram 89 89 89	Lalith
Sita 70 70 70	

Answer: (penalty regime: 0 %)

```

1 n=int(input())
2 d={}
3 for i in range(n):
4     s=input().split()
5     d[s[0]]=list(map(int,s[1:4]))
6 lname=[]
7 hname=[]
8 maxass=[]
9 minlab=[]
10 la=101

```

```

11 | ha=-1
12 | ma=-1
13 | ml=101
14 | for i,j in d.items():
15 |     avg=sum(j)/3
16 |     t,a,l=j
17 |     if(avg>ha):
18 |         ha=avg
19 |         hname=[i]
20 |     elif avg==ha:
21 |         hname.append(i)
22 |     if(avg<la):
23 |         la=avg
24 |         lname=[i]
25 |     elif avg==la:
26 |         lname.append(i)
27 |     if a>ma:
28 |         ma=a
29 |         maxass=[i]
30 |     elif(a==ma):
31 |         maxass.append(i)
32 |     if(l<ml):
33 |         ml=l
34 |         minlab=[i]
35 |     elif l==ml:
36 |         minlab.append(i)
37 | print(' '.join(sorted(hname)))
38 | print(' '.join(sorted(maxass)))
39 | print(' '.join(sorted(minlab)))
40 | print(' '.join(sorted(lname)))

```

	Input	Expected	Got	
✓	4 James 67 89 56 Lalith 89 45 45 Ram 89 89 89 Sita 70 70 70	Ram James Ram Lalith Lalith	Ram James Ram Lalith Lalith	✓
✓	3 Raja 95 67 90 Aarav 89 90 90 ShadhanA 95 95 91	ShadhanA ShadhanA Aarav Raja Raja	ShadhanA ShadhanA Aarav Raja Raja	✓

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

Question 10 | Correct Mark 1.00 out of 1.00

A company wants to send its quotation secretly to its client. The company decided to encrypt the amount they are sending to their client with some special symbols so that the equation amount will not be revealed to any external person. They used the special symbols !,@,#,\$,%,&,*,>,< for 0,1,2,3,4,5,6,7,8,9 respectively. Write a python code to help the company to convert the amount to special symbols.

(Value rounded off to 2 decimal points)

Input

n: Float data type which reads amount to send

Output

s: : String data type which displays symbols

Sample Testcase 1

Input

10000

Output

@!!!!!!

Sample Testcase2

1234.56

Output

@#\$%^.^&

For example:

Input	Result
1345.23	@\$%^.#\$
15000.59	@^!!!!.^<
156789	@^&*><.!!

Answer: (penalty regime: 0 %)

```

1 d={'0':'!', '1':'@', '2':'#', '3':'$', '4':'%', '5':'^', '6':'&', '7':'*', '8':'>', '9':'<'}
2 n=float(input().strip())
3 num_str=f"{n:.2f}"
4 encode=''.join(d.get(ch,ch) for ch in num_str)
5 print(encode)

```

	Input	Expected	Got	
✓	1345.23	@\$%^.#\$	@\$%^.#\$	✓
✓	15000.59	@^!!!.^<	@^!!!.^<	✓
✓	1234	@#\$\$%.!!	@#\$\$%.!!	✓
✓	156789	@^&*><.!!	@^&*><.!!	✓

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.