vl dharshini
231501038
aiml a

1) 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.

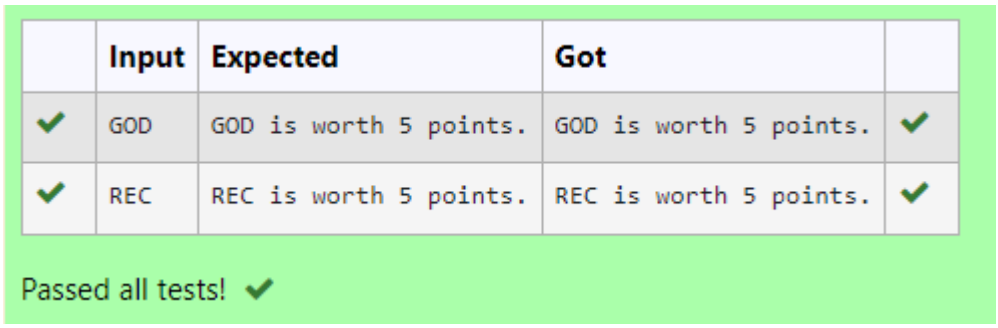
PROGRAM:

```
def scrabble_score(word):
    points = {
        'A': 1, 'E': 1, 'I': 1, 'L': 1, 'N': 1, 'O': 1, 'R': 1, 'S': 1, 'T': 1, 'U': 1,
        'D': 2, 'G': 2,
        'B': 3, 'C': 3, 'M': 3, 'P': 3,
        'F': 4, 'H': 4, 'V': 4, 'W': 4, 'Y': 4,
        'K': 5,
        'J': 8, 'X': 8,
        'Q': 10, 'Z': 10
    }
```

```
    score = sum(points.get(letter, 0) for letter in word.upper())


    return f"{word} is worth {score} points."
a=input()
print(scrabble_score(a))
```

OUTPUT:



	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! ✔

2) 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

PROGRAM:

```
def uncommon_words(s1, s2):
    words = s1.split() + s2.split()
    word_count = {}
    for word in words:
        word_count[word] = word_count.get(word, 0) + 1
        w=[word for word in word_count if word_count[word] == 1]
    return " ".join(w)
s1 =input()
s2 =input()
print(uncommon_words(s1, s2))
```

PROGRAM:

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

Passed all tests! ✔

3) 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

PROGRAM:

```
n=int(input())
a=[]
d={}
for i in range(n):
    a=input().split()
    sum=0
    name=a[0]
    nos=[int(x) for x in a[1:]]
    for i in nos:
        sum+=i
    d[name]=sum
    d= dict(sorted(d.items(), key=lambda item: item[1]))
    s = ''.join(f'{key} {value}\n' for key, value in d.items())
print(s)
```

OUTPUT:

	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! ✔

4) 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 Johny get maximum votes. Since John is alphabetically smaller, we print it.
Use dictionary to solve the above problem

Sample Input:

10

John

John

Johny

Jamie

Jamie

Johny

Jack

Johny

Johny

Jackie

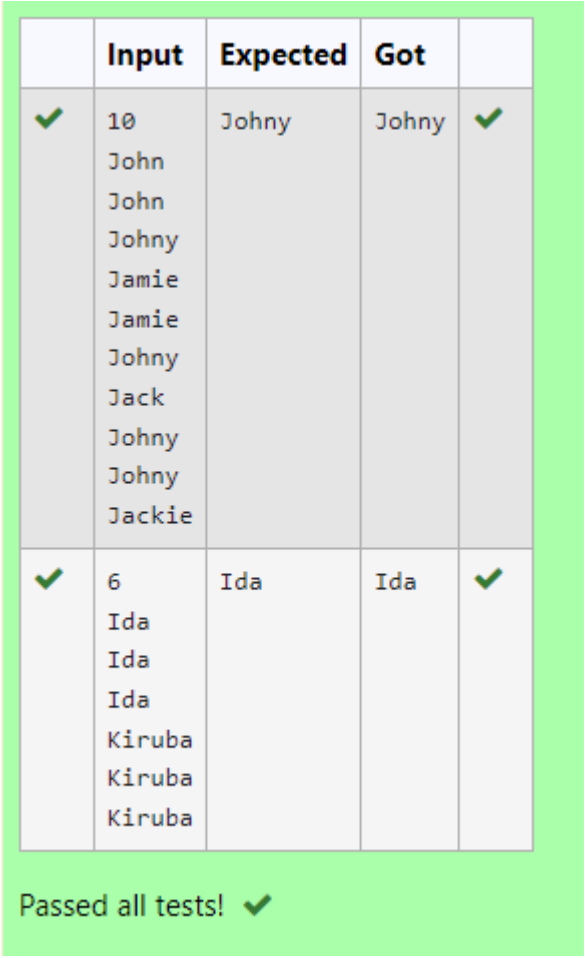
Sample Output:

Johny

PROGRAM:

```
def find_winner(votes):
    vote_count = {}
    for candidate in votes:
        if candidate in vote_count:
            vote_count[candidate] += 1
        else:
            vote_count[candidate] = 1
    max_votes = max(vote_count.values())
    max_vote_candidates = [candidate for candidate, votes in vote_count.items() if votes == max_votes]
    max_vote_candidates.sort()
    return max_vote_candidates[0]
num_votes = int(input())
votes = []
for i in range(num_votes):
    vote = input()
    votes.append(vote)
print(find_winner(votes))
```

OUTPUT:



	Input	Expected	Got	
✔	10 John John Johny Jamie Jamie Johny Jack Johny Johny Jackie	Johny	Johny	✔
✔	6 Ida Ida Ida Kiruba Kiruba Kiruba	Ida	Ida	✔

Passed all tests! ✔

5) 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 as 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

PROGRAM:

```
def highest_average_score(students):
    averages = {name: sum(scores) / len(scores) for name, scores in students.items()}
    max_average = max(averages.values())
    return sorted([name for name, avg in averages.items() if avg == max_average])


def highest_assignment_marks(students):
    max_assignment_marks = max((scores[1] for scores in students.values()))
    return sorted([name for name, scores in students.items() if scores[1] ==
max_assignment_marks])


def lowest_lab_marks(students):
    min_lab_marks = min((scores[2] for scores in students.values()))
    return sorted([name for name, scores in students.items() if scores[2] == min_lab_marks])


def lowest_average_score(students):
    averages = {name: sum(scores) / len(scores) for name, scores in students.items()}
    min_average = min(averages.values())
    return sorted([name for name, avg in averages.items() if avg == min_average])


# Get input from the user
num_students = int(input())
students_data = {}
```

```
for _ in range(num_students):
    student_info = input().split()
    name = student_info[0]
    marks = list(map(int, student_info[1:]))
    students_data[name] = marks

# Perform computations and print results
print(*highest_average_score(students_data))
print(*highest_assignment_marks(students_data))
print(*lowest_lab_marks(students_data))
print(*lowest_average_score(students_data))
```

OUTPUT:

	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! ✔