

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

Ex. No. : 9.1 Date:

Register No.: Name:

## Uncommon words

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.

#### PROGRAM:

```
aa=input()
a=aa.split(" ")
aaaa=[]
bb=input()
b=bb.split(" ")
for i in range(0, len(a)):
  if(a[i] not in aaaa):
      if(a.count(a[i])==1):
           if(a[i] not in b):
                print(a[i],end=" ")
                aaaa.append(a[i])
for j in range(0,len(b)):
   if(b[j] not in aaaa):
        if(b.count(b[j])==1):
           if(b[j] not in a):
                print(b[j],end=" ")
                aaaa.append(b[j])
```

 $\textbf{Input}: test\_dict = \{ \text{`Gfg'}: [6, \, 7, \, 4], \, \text{`best'}: [7, \, 6, \, 5] \}$ 

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

 $\ensuremath{\textbf{Explanation}}$  : Sorted by sum, and replaced.

 $Input : test\_dict = \{ Gfg' : [8,8], Gest' : [5,5] \}$ 

**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:

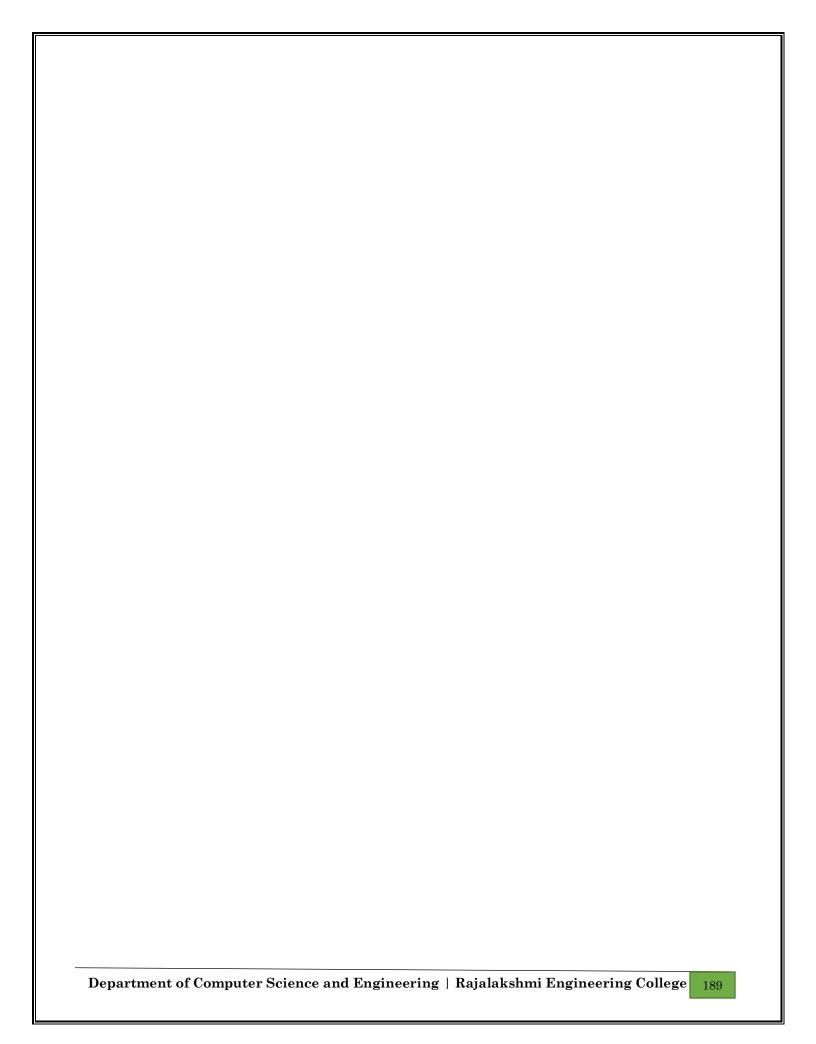
Input	Result
2 Gfg 6 7 4 Best 7 6 5	Gfg 17 Best 18

Ex. No.	:	9.2	Date:
Register No.:			Name:

# **Sort Dictionary by Values Summation**

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

#### PROGRAM:



### **Examples:**

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

#### For example:

r or champio.			
Input	Result		
John Johny Jamie Jamie Johny Jack Johny Johny	Johny		
Jackie			

Ex. No. : 9.3 Date:

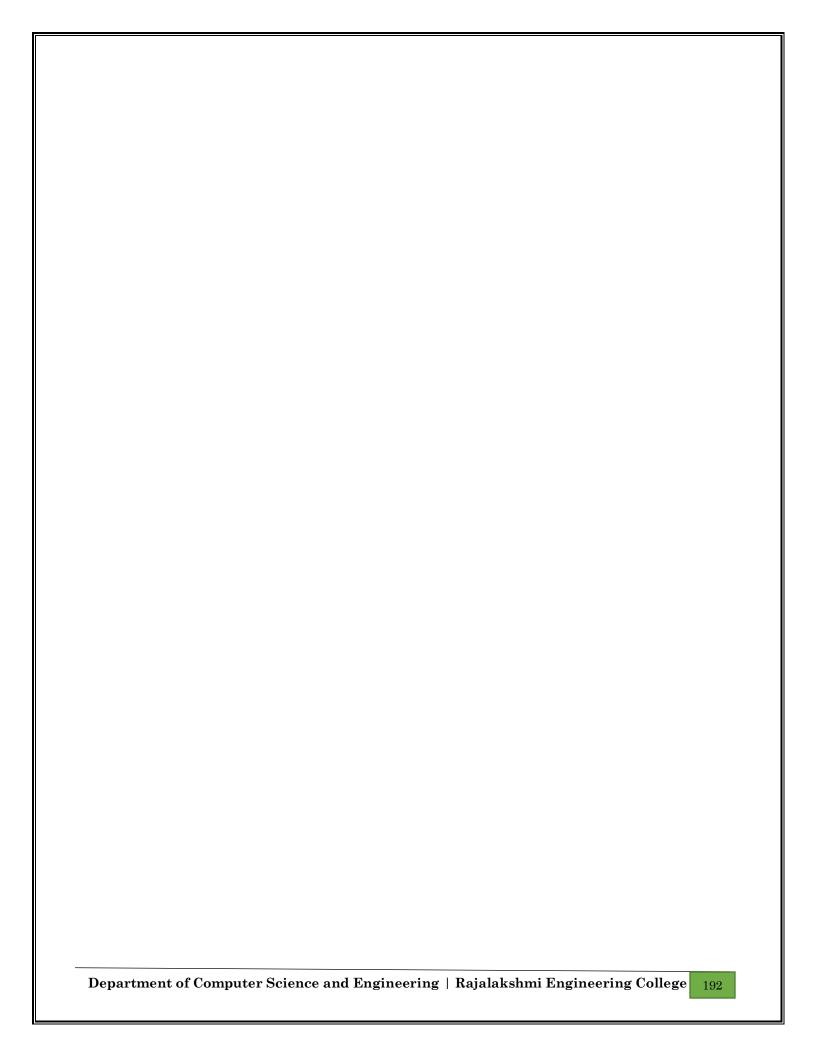
Register No.: Name:

## **Winner of Election**

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.

#### PROGRAM:

```
a=int(input())
aa=[]
b=[]
bbb=""
cc=0
for i in range(0,a):
    aaa=input()
    aa.append(aaa)
for j in range(0,len(aaa)):
    if(aa[j] not in b):
        if(int(aa.count(aa[j]))>(cc)):
        cc=aa.count(aa[j])
        bbb=aa[j]
        b.append(aa[j])
print(bbb)
```



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

Ex. No. : 9.4 Date:

Register No.: Name:

## **Student Record**

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

def process\_student\_marks():

4.Identify the student with the lowest average score

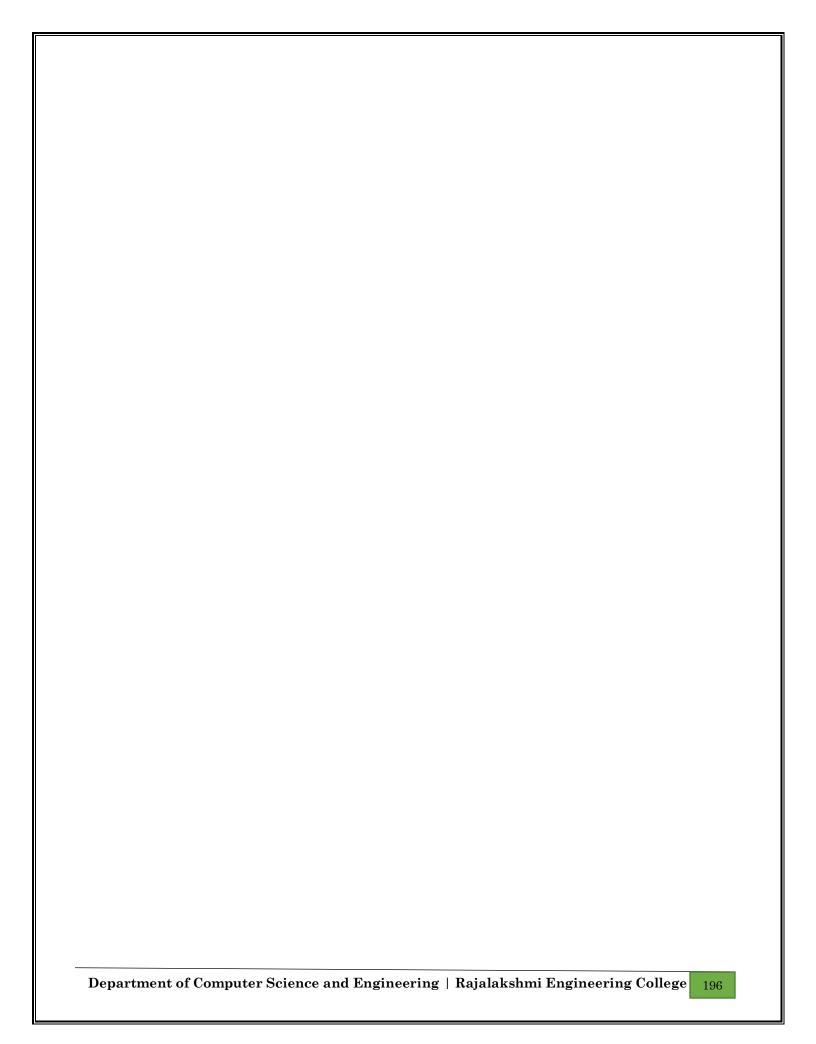
Note:

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

```
n = int(input(""))
students = {}

for _ in range(n):
    data = input().strip()
    name, test_mark, assignment_mark, lab_mark = data.split()
    students[name] = {
        'test_mark': int(test_mark),
        'assignment_mark': int(assignment_mark),
        'lab_mark': int(lab_mark)
        }
    averages = {name: (marks['test_mark'] + marks['assignment_mark'] + marks['lab_mark']) / 3
        for name, marks in students.items()}
```

```
max_average = max(averages.values())
            highest_avg_students = sorted([name for name, avg in averages.items()
if avg == max average
            max_assignment = max(students[name]['assignment_mark'] for name
in students)
            highest_assignment_students = sorted([name for name in students if
students[name]['assignment_mark'] == max_assignment])
            min lab = min(students[name]['lab mark'] for name in students)
            lowest_lab_students = sorted([name for name in students if
students[name]['lab_mark'] == min_lab])
            min_average = min(averages.values())
  lowest avg students = sorted([name for name, avg in averages.items() if avg ==
min_averagel)
            print(" ".join(highest_avg_students))
            print(" ".join(highest_assignment_students))
            print(" ".join(lowest_lab_students))
            print(" ".join(lowest_avg_students))
process_student_marks()
```



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

 $10~\mathrm{Q}$  and  $\mathrm{Z}$ 

Sample Input

REC

Sample Output

REC is worth 5 points.

Ex. No. : 9.5 Date:

Register No.: Name:

## **Scramble Score**

In the game of Scrabble<sup>TM</sup>, 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.

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

A Scrabble<sup>TM</sup> 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.

PROGRAM:

print("%s is worth %d points."%(a,sum))

```
aa=input()
g={"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}
a=""
sum=0
for i in aa:
    if(i in g):
        sum=sum+g[i]
        a=a+i
```

