Week-6 Graded Assignment (Programming)

Week-6 Graded Assignment (Programming)

```
Problem 1
   Question
   Prefix (Shown)
   Answer
   Suffix (Hidden)
   Test cases
       Public
        Private
Problem 2
   Question
   Prefix (Shown)
   Answer
   Suffix (Hidden)
   Test cases
       Public
        Private
   Tags
Problem 3
   Question
   Prefix (Shown)
   Answer
   Suffix (Hidden)
    Test cases
        Public
```

Private

Problem 1

Question

Write a function freqwords which accepts a list of words as a parameter and returns a dictionary which has the following structure:

- key: frequency of words in the list
- value: list of all words that have the above frequency

For example, consider this input:

```
1 ['No', 'sentence', 'can', 'begin', 'with', 'because', 'because', 'because', 'is', 'a', 'conjunction.']
```

The output dictionary corresponding to this is:

```
Key Value

['no', 'sentence', 'can', 'begin', 'with', 'is', 'a', 'conjunction']

['because']
```

Note

- All keys of the returned dictionary should be in lowercase. This means that words like 'It' and 'it' in the input list are equivalent.
- Remove any occurrence of these characters from all the words present in the list ,;:.!. These characters correspond to: comma, semi-colon, colon, full-stop, exclamation mark.
- You don't need to accept the input from the user or print the output to the console. This will be processed internally.
- You only need to fill the details in the body of the function.

Prefix (Shown)

```
malgudi = ['It', 'was', 'Monday', 'morning.', 'Swaminathan', 'was',
    'reluctant', 'to', 'open', 'his',

'eyes.', 'He', 'considered', 'Monday', 'specially', 'unpleasant', 'in',
    'the', 'calendar.', 'After',

'the', 'delicious', 'freedom', 'of', 'Saturday', 'And', 'Sunday,', 'it',
    'was', 'difficult', 'to',

'get', 'into', 'the', 'Monday', 'mood', 'of', 'work', 'and', 'discipline.',
    'He', 'shuddered', 'at',

'the', 'very', 'thought', 'of', 'school:', 'the', 'dismal', 'yellow',
    'building;', 'the',
    'fire-eyed', 'Vedanayagam,', 'his', 'class', 'teacher,', 'and',
    'headmaster', 'with', 'his',
    'thin', 'long', 'cane...']
```

Answer

```
def freqWords(wordList):
        freq = {} # dictionary to store word as key and frequenct as value
 3
        for wd in wordList:
            for i in ',;:.!': # iterating over the string contains the
    characters to be removed
                while i in wd: # do until no removeable left
                    wd = wd.replace(i, '')
 7
            if wd.lower() not in freq.keys(): # initialization of new words as
    keys
8
                freq[wd.lower()] = 0
9
            freq[wd.lower()] += 1
10
        freqWords_ = {} # dictionary to store frequency as key and word as value
11
        for wd in freq.keys():
            if freq[wd] not in freqWords_.keys(): # initialization for new
12
    frequency as key with empty list
13
                freqWords_[freq[wd]] = []
            freqWords_[freq[wd]].append(wd) # dictionary inversion
15
        return freqWords_
```

Suffix (Hidden)

```
1 | dataset = {}
    dataset['malgudi'] = malgudi
 3 dataset['A Red, Red Rose'] = '''O my Luve's like a red, red rose
    That's newly sprung in June;
    O my Luve's like the melodie
6
    That's sweetly play'd in tune.
 7
   As fair art thou, my bonnie lass,
9
    So deep in luve am I:
    And I will luve thee still, my dear,
10
    Till a' the seas gang dry:
11
12
    Till a' the seas gang dry, my dear,
13
    And the rocks melt wi' the sun:
14
    I will luve thee still, my dear,
15
    While the sands o' life shall run.
16
17
    And fare thee well, my only Luve
18
```

```
19 And fare thee well, a while!
20 And I will come again, my Luve,
    Tho' it were ten thousand mile.'''.strip().split()
21
22
23 dataset['Song of Myself'] = '''I celebrate myself, and sing myself,
24
      And what I assume you shall assume,
25
      For every atom belonging to me as good belongs to you.
26
27
      I loafe and invite my soul,
28
      I lean and loafe at my ease observing a spear of summer grass.
29
30
      My tongue, every atom of my blood, form'd from this soil, this air,
31
      Born here of parents born here from parents the same, and their
         parents the same,
32
33
      I, now thirty-seven years old in perfect health begin,
      Hoping to cease not till death.
34
35
36
      Creeds and schools in abeyance,
      Retiring back a while sufficed at what they are, but never forgotten,
37
      I harbor for good or bad, I permit to speak at every hazard,
      Nature without check with original energy.'''.strip().split()
39
40
41 | fw = freqWords(dataset[input().strip()])
42 for i in sorted(list(fw.keys())):
43
        print(i)
        print(*sorted(fw[i]))
44
```

Test cases

Public

Input	Output
malgudi	after at building calendar cane class considered delicious difficult discipline dismal eyes fire-eyed freedom get headmaster in into long mood morning open reluctant saturday school shuddered specially sunday swaminathan teacher thin thought unpleasant vedanayagam very with work yellow 2 he it to 3 and his monday of was 6 the

Private

Input	Output	
A Red, Red Rose	again am art as bonnie come deep fair it june lass life melodie melt mile newly only o' play'd rocks rose run sands shall so sprung sun sweetly ten thou thousand tho' tune were wi' 2 a a' dry fare gang like luve's o red seas still that's till well while 3 dear in will 4 i thee 5 and luve 6 the 8 my	
Song of Myself	abeyance air are as back bad begin belonging belongs blood but cease celebrate check creeds death ease energy forgotten form'd grass harbor hazard health hoping invite lean me nature never not now observing old or original perfect permit retiring schools shall sing soil soul speak spear sufficed summer their they thirty-seven till tongue while with without years 2 a assume atom born for from good here in loafe myself same the this what you 3 at every of parents 4 my to 6 and 7 i	

Problem 2

Question

An institution decides to allow students to create student groups for each subject where students with similar marks can help each other. But it draws up a set of constraints for creating student groups:

- A group is associated with a particular subject.
- The group can consist of any number of students.
- The maximum difference of marks in this subject between the highest and lowest scorer within the group should be markLimit.

The **Scores Dataset** from the CT course will have the student details. Write a function called crowdedGroup that accepts three parameters as input:

- scores
- subject
- markLimit

The function should return a list of groups, where each group is a list of **SeqNo** of students of the largest possible group in that subject.

```
def crowdedGroup(scores, subject, markLimit):
1
        1.1.1
2
3
            Input:
                scores: list of dictionaries; Scores Dataset
                subject: string
                markLimit: integer
6
7
          Output: list of lists
8
                    inner list: list of integers (SegNo)
                    each inner list represents a group
9
10
```

Note

- scores is a list of dictionaries that represents the **Scores Dataset**. Look at the prefix code shown below to get an idea of the structure of scores.
- You don't need to accept the input from the user or print the output to the console. This will be processed internally.
- You only need to fill the details in the body of the function.

Prefix (Shown)

```
1
   scores = [ { 'Chemistry': 78,
2
       'CityTown': 'Erode',
3
       'DateOfBirth': '7 Nov',
       'Gender': 'M',
4
5
       'Mathematics': 68,
6
       'Name': 'Bhuvanesh',
       'Physics': 64,
       'SeqNo': 0,
8
9
        'Total': 210},
```

```
10
11
      { 'Chemistry': 91,
12
         'CityTown': 'Salem',
13
        'DateOfBirth': '3 Jun',
14
        'Gender': 'M',
15
        'Mathematics': 62,
         'Name': 'Harish',
16
17
         'Physics': 45,
18
        'SeqNo': 1,
19
        'Total': 198},
20
21
      { 'Chemistry': 77,
22
         'CityTown': 'Chennai',
23
        'DateOfBirth': '4 Jan',
24
         'Gender': 'M',
25
        'Mathematics': 57,
26
         'Name': 'Shashank',
27
         'Physics': 54,
        'SeqNo': 2,
28
29
        'Total': 188},
30
31
      { 'Chemistry': 78,
32
         'CityTown': 'Chennai',
33
        'DateOfBirth': '5 May',
34
         'Gender': 'F',
        'Mathematics': 42,
35
         'Name': 'Rida',
36
37
         'Physics': 53,
38
        'SeqNo': 3,
39
         'Total': 173},
40
41
      { 'Chemistry': 89,
         'CityTown': 'Madurai',
42
        'DateOfBirth': '17 Nov',
43
44
         'Gender': 'F',
45
        'Mathematics': 87,
         'Name': 'Ritika',
46
        'Physics': 64,
47
        'SeqNo': 4,
48
49
        'Total': 240},
50
      { 'Chemistry': 84,
51
52
        'CityTown': 'Chennai',
        'DateOfBirth': '8 Feb',
53
        'Gender': 'F',
54
55
        'Mathematics': 71,
         'Name': 'Akshaya',
56
57
        'Physics': 92,
        'SeqNo': 5,
58
59
        'Total': 247},
60
61
      { 'Chemistry': 87,
62
        'CityTown': 'Ambur',
         'DateOfBirth': '23 Mar',
63
64
         'Gender': 'M',
65
         'Mathematics': 81,
66
         'Name': 'Sameer',
        'Physics': 82,
67
```

```
68
          'SeqNo': 6,
          'Total': 250},
 69
 70
        { 'Chemistry': 76,
 71
          'CityTown': 'Vellore',
 72
 73
          'DateOfBirth': '15 Mar',
 74
          'Gender': 'M',
          'Mathematics': 84,
 75
 76
          'Name': 'Aditya',
 77
          'Physics': 92,
          'SeqNo': 7,
 78
 79
          'Total': 252},
 80
 81
        { 'Chemistry': 51,
          'CityTown': 'Bengaluru',
 82
          'DateOfBirth': '28 Feb',
 83
 84
          'Gender': 'M',
 85
          'Mathematics': 74,
          'Name': 'Surya',
 86
          'Physics': 64,
 87
          'SeqNo': 8,
 88
          'Total': 189},
 89
 90
        { 'Chemistry': 73,
 91
 92
          'CityTown': 'Bengaluru',
          'DateOfBirth': '6 Dec',
 93
          'Gender': 'M',
 94
 95
          'Mathematics': 63,
          'Name': 'Clarence',
 96
 97
          'Physics': 88,
          'SeqNo': 9,
 98
 99
          'Total': 224},
100
        { 'Chemistry': 68,
101
102
          'CityTown': 'Chennai',
          'DateOfBirth': '12 Jan',
103
104
          'Gender': 'F',
          'Mathematics': 64,
105
          'Name': 'Kavya',
106
107
          'Physics': 72,
108
          'SeqNo': 10,
          'Total': 204},
109
110
        { 'Chemistry': 92,
111
112
          'CityTown': 'Bengaluru',
          'DateOfBirth': '30 Apr',
113
114
          'Gender': 'M',
115
          'Mathematics': 97,
          'Name': 'Rahul',
116
117
          'Physics': 92,
          'SeqNo': 11,
118
119
          'Total': 281},
120
        { 'Chemistry': 71,
121
122
          'CityTown': 'Chennai',
          'DateOfBirth': '14 Jan',
123
124
          'Gender': 'F',
          'Mathematics': 52,
125
```

```
126
          'Name': 'Srinidhi',
127
          'Physics': 64,
          'SeqNo': 12,
128
129
          'Total': 187},
130
        { 'Chemistry': 89,
131
132
          'CityTown': 'Madurai',
          'DateOfBirth': '6 May',
133
134
          'Gender': 'M',
135
          'Mathematics': 65,
          'Name': 'Gopi',
136
137
          'Physics': 73,
138
          'SeqNo': 13,
139
          'Total': 227},
140
        { 'Chemistry': 93,
141
142
          'CityTown': 'Trichy',
          'DateOfBirth': '23 July',
143
          'Gender': 'F',
144
145
          'Mathematics': 89,
          'Name': 'Sophia',
146
          'Physics': 62,
147
148
          'SeqNo': 14,
          'Total': 244},
149
150
        { 'Chemistry': 90,
151
          'CityTown': 'Theni',
152
153
          'DateOfBirth': '22 Sep',
          'Gender': 'F',
154
155
          'Mathematics': 76,
          'Name': 'Goutami',
156
157
          'Physics': 58,
          'SeqNo': 15,
158
159
          'Total': 224},
160
161
        { 'Chemistry': 43,
          'CityTown': 'Trichy',
162
          'DateOfBirth': '30 Dec',
163
          'Gender': 'M',
164
165
          'Mathematics': 87,
          'Name': 'Tauseef',
166
167
          'Physics': 86,
168
          'SeqNo': 16,
169
          'Total': 216},
170
        { 'Chemistry': 67,
171
172
          'CityTown': 'Chennai',
173
          'DateOfBirth': '14 Dec',
          'Gender': 'M',
174
175
          'Mathematics': 62,
          'Name': 'Arshad',
176
177
          'Physics': 81,
178
          'SeqNo': 17,
          'Total': 210},
179
180
181
        { 'Chemistry': 97,
182
          'CityTown': 'Erode',
183
          'DateOfBirth': '9 Oct',
```

```
184
          'Gender': 'F',
185
          'Mathematics': 72,
186
          'Name': 'Abirami',
187
          'Physics': 92,
188
          'SeqNo': 18,
189
          'Total': 261},
190
        { 'Chemistry': 62,
191
192
          'CityTown': 'Trichy',
193
          'DateOfBirth': '30 Aug',
          'Gender': 'M',
194
195
          'Mathematics': 56,
          'Name': 'Vetrivel',
196
197
          'Physics': 78,
198
          'SeqNo': 19,
          'Total': 196},
199
200
201
        { 'Chemistry': 91,
          'CityTown': 'Vellore',
202
203
          'DateOfBirth': '17 Sep',
          'Gender': 'M',
204
205
          'Mathematics': 93,
206
          'Name': 'Kalyan',
          'Physics': 68,
207
208
          'SeqNo': 20,
          'Total': 252},
209
210
        { 'Chemistry': 74,
211
          'CityTown': 'Bengaluru',
212
213
          'DateOfBirth': '15 Mar',
          'Gender': 'F',
214
          'Mathematics': 78,
215
          'Name': 'Monika',
216
217
          'Physics': 69,
218
          'SeqNo': 21,
219
          'Total': 221},
220
221
        { 'Chemistry': 57,
          'CityTown': 'Nagercoil',
222
223
          'DateOfBirth': '17 Jul',
224
          'Gender': 'F',
225
          'Mathematics': 62,
226
          'Name': 'Priya',
          'Physics': 62,
227
228
          'SeqNo': 22,
          'Total': 181},
229
230
231
        { 'Chemistry': 88,
          'CityTown': 'Bengaluru',
232
233
          'DateOfBirth': '13 May',
          'Gender': 'F',
234
235
          'Mathematics': 97,
236
          'Name': 'Deepika',
          'Physics': 91,
237
238
          'SeqNo': 23,
239
          'Total': 276},
240
        { 'Chemistry': 58,
241
```

```
242
          'CityTown': 'Madurai',
243
          'DateOfBirth': '26 Dec',
244
          'Gender': 'M',
245
          'Mathematics': 44,
246
          'Name': 'Siddharth',
247
          'Physics': 72,
248
          'SeqNo': 24,
          'Total': 174},
249
250
251
        { 'Chemistry': 92,
252
          'CityTown': 'Chennai',
253
          'DateOfBirth': '16 May',
254
          'Gender': 'F',
255
          'Mathematics': 87,
          'Name': 'Geeta',
256
          'Physics': 75,
257
258
          'SeqNo': 25,
259
          'Total': 254},
260
261
        { 'Chemistry': 82,
          'CityTown': 'Chennai',
262
          'DateOfBirth': '22 Jul',
263
264
          'Gender': 'M',
          'Mathematics': 74,
265
266
          'Name': 'JK',
          'Physics': 71,
267
          'SeqNo': 26,
268
269
          'Total': 227},
270
271
        { 'Chemistry': 52,
272
          'CityTown': 'Madurai',
          'DateOfBirth': '4 Mar',
273
          'Gender': 'M',
274
275
          'Mathematics': 81,
276
          'Name': 'Jagan',
277
          'Physics': 76,
          'SeqNo': 27,
278
          'Total': 209},
279
280
281
        { 'Chemistry': 83,
282
          'CityTown': 'Madurai',
          'DateOfBirth': '10 Sep',
283
284
          'Gender': 'F',
285
          'Mathematics': 74,
286
          'Name': 'Nisha',
          'Physics': 83,
287
288
          'SeqNo': 28,
289
          'Total': 240},
290
291
        { 'Chemistry': 81,
          'CityTown': 'Vellore',
292
293
          'DateOfBirth': '13 Oct',
294
          'Gender': 'M',
          'Mathematics': 72,
295
296
          'Name': 'Naveen',
297
          'Physics': 66,
298
          'SeqNo': 29,
299
          'Total': 219}]
```

Answer

```
def crowdedGroup(scores, sub, markLimit):
 2
        subDict = {} # dictionary to store subject mark as key and list of
    students' ID as value
 3
        for sDict in scores:
 4
            if sDict[sub] not in subDict.keys():
                subDict[sDict[sub]] = [] # Initialization for new keys
                subDict[sDict[sub]].append(sDict['SeqNo']) # Append the student
    ID inside the list in key of subject marks
 7
            else:
 8
                subDict[sDict[sub]].append(sDict['SeqNo'])
        maxNumber, crowdedGroup_ = 0, [] # variable to hold maximum group count
    and list of IDs of group
        for i in range(0, 101-markLimit): # Iterating from zero to maximum mark
10
    - limit
            group = []
11
            for m in subDict.keys(): # Iterating through each marks
12
13
                if i <= m <= i+markLimit:</pre>
14
                     group.extend(subDict[m]) # Append all the IDs if it is in
    the valid limit
            if len(group) > maxNumber:
15
                maxNumber = len(group) # Updating the maximum group count so far
16
                crowedGroup_ = [group] # Updating the maximum group list so far
17
18
            if len(group) == maxNumber and group not in crowedGroup_:
                crowedGroup_.append(group) # adding another group having same
19
    count
20
        return crowdedGroup_
```

Suffix (Hidden)

```
sub, markLimit = input().strip().split()
markLimit = int(markLimit)
members = crowdedGroup(scores, sub, markLimit)
for m in members:
print(*sorted(m))
```

Test cases

Public

Input	Output			
Dhygias 25	0 4 6 8 10 12 13 14 15 17 19 20 21 22 24 25 26 27 28 29			
Physics 25	0 4 6 8 10 12 13 14 16 17 19 20 21 22 24 25 26 27 28 29			

Private

Input	Output
Chemistry 30	0 1 2 3 4 5 6 7 9 10 11 12 13 14 15 17 18 20 21 23 25 26 28 29
Mathematics	0 1 2 4 5 6 7 8 9 10 13 15 16 17 18 21 22 25 26 27 28 29 0 1 4 5 6 7 8 9 10 13 14 15 16 17 18 21 22 25 26 27 28 29
Physics 32	0 4 5 6 7 8 9 10 11 12 13 14 16 17 18 19 20 21 22 23 24 25 26 27 28 29

Tags

Problem 3

Question

The **Scores Dataset** from the CT course has the details of a class of students. A student X can mentor another student Y in **subject** if the following conditions are satisfied:

- X has scored more than Y in **subject**.
- The difference in marks between X and Y in **subject** lies in the range [10, 20], both end points included.

Write a function topMentors that accepts the following parameters as input:

- scores
- subject

The function should return a dictionary having the following structure as output:

- key: SeqNo of the student who can mentor the largest number of students in subject
- value: list of SeqNo of students who can be mentored by the above student

```
def topMentors(scores, subject):
1
         1 \cdot 1 \cdot 1
2
3
             Input:
                  scores: list of dictionaries; Scores dataset
4
                  subject: string
5
6
             Output:
7
                 dictionary:
8
                      key: integer (SeqNo)
9
                      value: list of integers (SeqNo)
         1.1.1
10
```

Note

- scores is a list of dictionaries that represents the **Scores Dataset**. Look at the prefix code shown below to get an idea of the structure of scores.
- You don't need to accept the input from the user or print the output to the console. This will be processed internally.
- You only need to fill the details in the body of the function.

Prefix (Shown)

```
1
    scores = [ { 'Chemistry': 78,
2
        'CityTown': 'Erode',
3
        'DateOfBirth': '7 Nov',
4
        'Gender': 'M',
5
        'Mathematics': 68,
        'Name': 'Bhuvanesh',
 6
        'Physics': 64,
 7
        'SeqNo': 0,
8
9
        'Total': 210},
10
11
      { 'Chemistry': 91,
12
        'CityTown': 'Salem',
```

```
13
         'DateOfBirth': '3 Jun',
14
         'Gender': 'M',
15
         'Mathematics': 62,
16
        'Name': 'Harish',
17
         'Physics': 45,
         'SeqNo': 1,
18
19
         'Total': 198},
20
21
      { 'Chemistry': 77,
22
         'CityTown': 'Chennai',
        'DateOfBirth': '4 Jan',
23
24
         'Gender': 'M',
         'Mathematics': 57,
25
26
        'Name': 'Shashank',
         'Physics': 54,
27
         'SeqNo': 2,
28
29
         'Total': 188},
30
31
      { 'Chemistry': 78,
         'CityTown': 'Chennai',
32
        'DateOfBirth': '5 May',
33
34
         'Gender': 'F',
35
         'Mathematics': 42,
36
        'Name': 'Rida',
         'Physics': 53,
37
        'SeqNo': 3,
38
         'Total': 173},
39
40
      { 'Chemistry': 89,
41
42
         'CityTown': 'Madurai',
        'DateOfBirth': '17 Nov',
43
         'Gender': 'F',
44
45
         'Mathematics': 87,
        'Name': 'Ritika',
46
47
         'Physics': 64,
         'SeqNo': 4,
48
49
         'Total': 240},
50
      { 'Chemistry': 84,
51
52
         'CityTown': 'Chennai',
53
         'DateOfBirth': '8 Feb',
         'Gender': 'F',
54
55
        'Mathematics': 71,
        'Name': 'Akshaya',
56
57
        'Physics': 92,
58
         'SeqNo': 5,
59
         'Total': 247},
60
      { 'Chemistry': 87,
61
62
         'CityTown': 'Ambur',
         'DateOfBirth': '23 Mar',
63
64
         'Gender': 'M',
        'Mathematics': 81,
65
        'Name': 'Sameer',
66
67
        'Physics': 82,
68
         'SeqNo': 6,
69
         'Total': 250},
70
```

```
71
        { 'Chemistry': 76,
 72
          'CityTown': 'Vellore',
 73
          'DateOfBirth': '15 Mar',
 74
          'Gender': 'M',
 75
          'Mathematics': 84,
 76
          'Name': 'Aditya',
 77
          'Physics': 92,
 78
          'SeqNo': 7,
 79
          'Total': 252},
 80
        { 'Chemistry': 51,
 81
 82
          'CityTown': 'Bengaluru',
          'DateOfBirth': '28 Feb',
 83
 84
          'Gender': 'M',
 85
          'Mathematics': 74,
          'Name': 'Surya',
 86
 87
          'Physics': 64,
 88
          'SeqNo': 8,
          'Total': 189},
 89
 90
        { 'Chemistry': 73,
 91
          'CityTown': 'Bengaluru',
 92
 93
          'DateOfBirth': '6 Dec',
          'Gender': 'M',
 94
 95
          'Mathematics': 63,
          'Name': 'Clarence',
 96
          'Physics': 88,
 97
          'SeqNo': 9,
 98
          'Total': 224},
 99
100
        { 'Chemistry': 68,
101
          'CityTown': 'Chennai',
102
          'DateOfBirth': '12 Jan',
103
          'Gender': 'F',
104
105
          'Mathematics': 64,
          'Name': 'Kavya',
106
          'Physics': 72,
107
          'SeqNo': 10,
108
          'Total': 204},
109
110
        { 'Chemistry': 92,
111
          'CityTown': 'Bengaluru',
112
113
          'DateOfBirth': '30 Apr',
          'Gender': 'M',
114
115
          'Mathematics': 97,
          'Name': 'Rahul',
116
117
          'Physics': 92,
118
          'SeqNo': 11,
          'Total': 281},
119
120
        { 'Chemistry': 71,
121
122
          'CityTown': 'Chennai',
          'DateOfBirth': '14 Jan',
123
          'Gender': 'F',
124
125
          'Mathematics': 52,
          'Name': 'Srinidhi',
126
127
          'Physics': 64,
          'SeqNo': 12,
128
```

```
129
          'Total': 187},
130
        { 'Chemistry': 89,
131
132
          'CityTown': 'Madurai',
133
          'DateOfBirth': '6 May',
134
          'Gender': 'M',
135
          'Mathematics': 65,
          'Name': 'Gopi',
136
137
          'Physics': 73,
138
          'SeqNo': 13,
          'Total': 227},
139
140
141
        { 'Chemistry': 93,
142
          'CityTown': 'Trichy',
          'DateOfBirth': '23 July',
143
          'Gender': 'F',
144
145
          'Mathematics': 89,
146
          'Name': 'Sophia',
          'Physics': 62,
147
148
          'SeqNo': 14,
          'Total': 244},
149
150
151
        { 'Chemistry': 90,
          'CityTown': 'Theni',
152
153
          'DateOfBirth': '22 Sep',
          'Gender': 'F',
154
          'Mathematics': 76,
155
          'Name': 'Goutami',
156
          'Physics': 58,
157
158
          'SeqNo': 15,
          'Total': 224},
159
160
        { 'Chemistry': 43,
161
          'CityTown': 'Trichy',
162
163
          'DateOfBirth': '30 Dec',
164
          'Gender': 'M',
165
          'Mathematics': 87,
          'Name': 'Tauseef',
166
          'Physics': 86,
167
168
          'SeqNo': 16,
          'Total': 216},
169
170
171
        { 'Chemistry': 67,
          'CityTown': 'Chennai',
172
173
          'DateOfBirth': '14 Dec',
          'Gender': 'M',
174
175
          'Mathematics': 62,
176
          'Name': 'Arshad',
          'Physics': 81,
177
178
          'SeqNo': 17,
          'Total': 210},
179
180
        { 'Chemistry': 97,
181
          'CityTown': 'Erode',
182
183
          'DateOfBirth': '9 Oct',
          'Gender': 'F',
184
185
          'Mathematics': 72,
186
          'Name': 'Abirami',
```

```
187
          'Physics': 92,
188
          'SeqNo': 18,
189
          'Total': 261},
190
191
        { 'Chemistry': 62,
192
          'CityTown': 'Trichy',
193
          'DateOfBirth': '30 Aug',
          'Gender': 'M',
194
195
          'Mathematics': 56,
196
          'Name': 'Vetrivel',
          'Physics': 78,
197
198
          'SeqNo': 19,
          'Total': 196},
199
200
201
        { 'Chemistry': 91,
202
          'CityTown': 'Vellore',
203
          'DateOfBirth': '17 Sep',
204
          'Gender': 'M',
          'Mathematics': 93,
205
          'Name': 'Kalyan',
206
          'Physics': 68,
207
          'SeqNo': 20,
208
209
          'Total': 252},
210
211
        { 'Chemistry': 74,
          'CityTown': 'Bengaluru',
212
          'DateOfBirth': '15 Mar',
213
          'Gender': 'F',
214
          'Mathematics': 78,
215
216
          'Name': 'Monika',
          'Physics': 69,
217
218
          'SeqNo': 21,
          'Total': 221},
219
220
221
        { 'Chemistry': 57,
222
          'CityTown': 'Nagercoil',
          'DateOfBirth': '17 Jul',
223
          'Gender': 'F',
224
          'Mathematics': 62,
225
226
          'Name': 'Priya',
227
          'Physics': 62,
228
          'SeqNo': 22,
229
          'Total': 181},
230
231
        { 'Chemistry': 88,
          'CityTown': 'Bengaluru',
232
233
          'DateOfBirth': '13 May',
234
          'Gender': 'F',
235
          'Mathematics': 97,
236
          'Name': 'Deepika',
          'Physics': 91,
237
238
          'SeqNo': 23,
239
          'Total': 276},
240
241
        { 'Chemistry': 58,
          'CityTown': 'Madurai',
242
243
          'DateOfBirth': '26 Dec',
          'Gender': 'M',
244
```

```
245
          'Mathematics': 44,
246
          'Name': 'Siddharth',
247
          'Physics': 72,
248
          'SeqNo': 24,
249
          'Total': 174},
250
251
        { 'Chemistry': 92,
          'CityTown': 'Chennai',
252
253
          'DateOfBirth': '16 May',
254
          'Gender': 'F',
          'Mathematics': 87,
255
256
          'Name': 'Geeta',
257
          'Physics': 75,
258
          'SeqNo': 25,
          'Total': 254},
259
260
261
        { 'Chemistry': 82,
262
          'CityTown': 'Chennai',
          'DateOfBirth': '22 Jul',
263
264
          'Gender': 'M',
          'Mathematics': 74,
265
          'Name': 'JK',
266
267
          'Physics': 71,
          'SeqNo': 26,
268
269
          'Total': 227},
270
        { 'Chemistry': 52,
271
          'CityTown': 'Madurai',
272
          'DateOfBirth': '4 Mar',
273
274
          'Gender': 'M',
          'Mathematics': 81,
275
          'Name': 'Jagan',
276
          'Physics': 76,
277
278
          'SeqNo': 27,
279
          'Total': 209},
280
281
        { 'Chemistry': 83,
282
          'CityTown': 'Madurai',
          'DateOfBirth': '10 Sep',
283
284
          'Gender': 'F',
285
          'Mathematics': 74,
          'Name': 'Nisha',
286
287
          'Physics': 83,
288
          'SeqNo': 28,
289
          'Total': 240},
290
291
        { 'Chemistry': 81,
292
          'CityTown': 'Vellore',
          'DateOfBirth': '13 Oct',
293
294
          'Gender': 'M',
295
          'Mathematics': 72,
296
          'Name': 'Naveen',
          'Physics': 66,
297
          'SeqNo': 29,
298
299
          'Total': 219}]
```

Answer

```
def adjMat(scores, sub): # Function to find adjacency matrix
 2
        adj = {} # adjacency matrix as nested dictionary
 3
        for mentor in scores: # Outer dictionary / Row for mentor
            adj[mentor['SeqNo']] = {} # initializing with an empty dictionary
 4
    for all SeqNo
 5
            for mentored in scores: # Inner dictionary / column for mentored
                # mentoring condition and prevention of same student comparison
 6
                if 10 <= mentor[sub] - mentored[sub] <= 20 and mentor['SeqNo']</pre>
    != mentored['SeqNo']:
                    adj[mentor['SeqNo']][mentored['SeqNo']] = 1
 8
 9
        return adj
10
11
    def topMentors(scores, sub):
        adj = adjMat(scores, sub) # getting an adjacency matrix
12
        topMentors_ = {} # dictionary to store mentor as key and mentored SeqNo
13
    list as value
        maxMentored = 0 # value to hold the maximum mentored students
14
        for i in adj.keys():
            if len(adj[i]) > maxMentored:
16
                maxMentored = len(adj[i]) # replacing the maximum mentored count
17
18
                topMentors_ = {} # reinitialization with empty dictionary
19
                topMentors_[i] = list(adj[i].keys()) # adding entry with mentor
    as key and mentored list value
20
            if len(adj[i]) == maxMentored:
21
                topMentors_[i] = list(adj[i].keys()) # adding entry with mentor
    as key and mentored list value
       return topMentors_
22
```

Suffix (Hidden)

```
1  tm = topMentors(scores, input())
2  for i in sorted(list(tm.keys())):
3     print(i)
4     print(*sorted(tm[i]))
```

Test cases

Public

Input	Output		
Physics	6		
111,0200	0 4 8 10 12 14 20 21 22 24 26 29		

Private

Input	Output
Chemistry	14 0 2 3 7 9 21 26 28 29
Mathematics	7 0 5 8 10 13 18 26 28 29