



NUST School of Mechanical and Manufacturing Engineering (SMME)

Department of Robotics and Artificial Intelligence

RIME - 23

Artificial Intelligence CSE – 860

Assignment # 3

Submitted by: Fatima Nadeem

Registration #: 482575

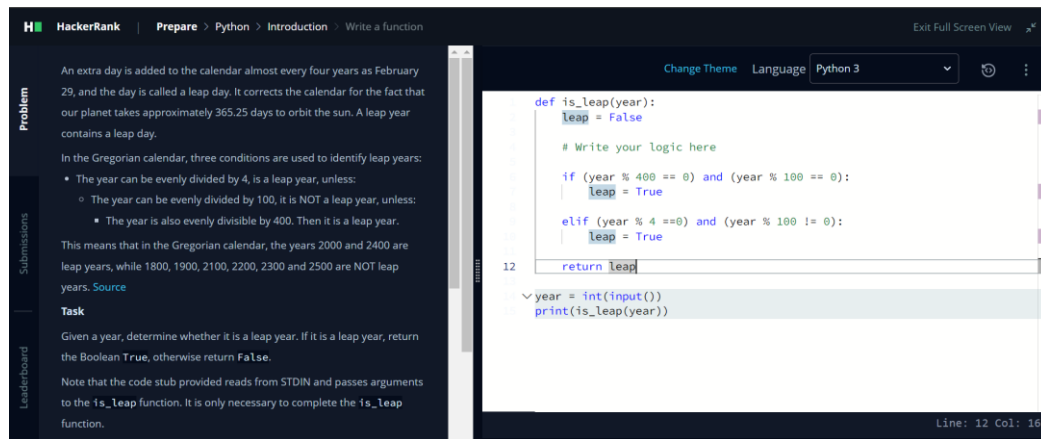
Submitted to: Dr. Yasar Ayaz

Assignment Prompt: Complete medium and hard challenges of Python on www.hackerrank.com

Medium Level Tasks:

1. Write a Function

Task and Code:



Problem

An extra day is added to the calendar almost every four years as February 29, and the day is called a leap day. It corrects the calendar for the fact that our planet takes approximately 365.25 days to orbit the sun. A leap year contains a leap day.

In the Gregorian calendar, three conditions are used to identify leap years:

- The year can be evenly divided by 4, is a leap year, unless:
 - The year can be evenly divided by 100, it is NOT a leap year, unless:
 - The year is also evenly divisible by 400. Then it is a leap year.

This means that in the Gregorian calendar, the years 2000 and 2400 are leap years, while 1800, 1900, 2100, 2200, 2300 and 2500 are NOT leap years. [Source](#)

Task

Given a year, determine whether it is a leap year. If it is a leap year, return the Boolean True, otherwise return False.

Note that the code stub provided reads from STDIN and passes arguments to the `is_leap` function. It is only necessary to complete the `is_leap` function.

```
def is_leap(year):
    leap = False

    # Write your logic here

    if (year % 400 == 0) and (year % 100 == 0):
        leap = True

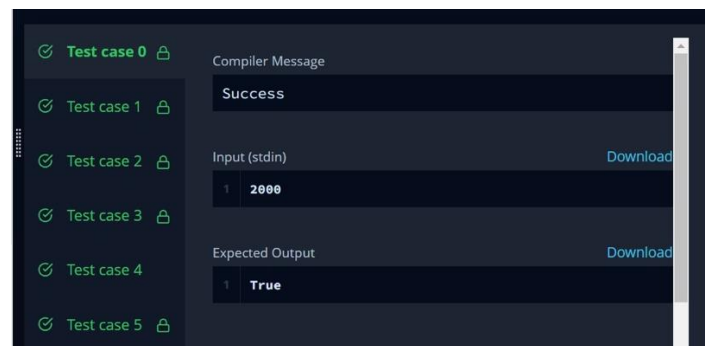
    elif (year % 4 == 0) and (year % 100 != 0):
        leap = True

    return leap

year = int(input())
print(is_leap(year))
```

Line: 12 Col: 16

Test Cases:



Test case 0 ✓

Test case 1 ✓

Test case 2 ✓

Test case 3 ✓

Test case 4 ✓

Test case 5 ✓

Compiler Message

Success

Input (stdin)

1 2000

Expected Output

1 True

2. The Minion Game

Task:

Kevin and Stuart want to play the 'The Minion Game'.

Game Rules

Both players are given the same string, *S*.

Both players have to make substrings using the letters of the string *S*.

Stuart has to make words starting with consonants.

Kevin has to make words starting with vowels.

The game ends when both players have made all possible substrings.

Scoring

A player gets +1 point for each occurrence of the substring in the string *S*.

For Example:

String *S* = BANANA

Kevin's vowel beginning word = ANA

Here, ANA occurs twice in BANANA. Hence, Kevin will get 2 Points.

Your task is to determine the winner of the game and their score.

Function Description

Complete the `minion_game` in the editor below.

`minion_game` has the following parameters:

- string *string*: the string to analyze

Prints

- string: the winner's name and score, separated by a space on one line, or **Draw** if there is no winner

Code:

```
1 def minion_game(string):
2     # your code goes here
3     vowel = 'aeiou'.upper()
4     strl = len(string)
5     kevin = sum(strl-i for i in range(strl) if string[i] in vowel)
6     stuart = strl*(strl + 1)/2 - kevin
7
8     if kevin == stuart:
9         print('Draw')
10
11    elif kevin > stuart:
12        print('Kevin %d' % kevin)
13
14    else:
15        print('Stuart %d' % stuart)
16
17 if __name__ == '__main__':
18     s = input()
19     minion_game(s)
```

Line: 16 Col: 1

Test Cases:

Test case 0 ✓

Test case 1 ✓

Test case 2 ✓

Test case 3 ✓

Test case 4 ✓

Test case 5 ✓

Test case 6 ✓

Compiler Message

Success

Input (stdin) Download

1 BANANA

Expected Output Download

1 Stuart 12

3. Merge the Tools!

Task:

Problem	Consider the following:
	<ul style="list-style-type: none">A string, s, of length n where $s = c_0c_1 \dots c_{n-1}$.An integer, k, where k is a factor of n.
	We can split s into $\frac{n}{k}$ substrings where each substring, t_i , consists of a contiguous block of k characters in s . Then, use each t_i to create string u_i such that:
	<ul style="list-style-type: none">The characters in u_i are a subsequence of the characters in t_i.Any repeat occurrence of a character is removed from the string such that each character in u_i occurs exactly once. In other words, if the character at some index j in t_i occurs at a previous index $< j$ in t_i, then do not include the character in string u_i.
Submissions	Given s and k , print $\frac{n}{k}$ lines where each line i denotes string u_i .
	Example
	$s = \text{'AAABCADDE'}$
	$k = 3$
Leaderboard	There are three substrings of length 3 to consider: 'AAA', 'BCA' and 'DDE'. The first substring is all 'A' characters, so $u_1 = \text{'A'}$. The second substring has all distinct characters, so $u_2 = \text{'BCA'}$. The third substring has 2 different characters, so $u_3 = \text{'DE'}$. Note that a subsequence maintains the original order of characters encountered. The order of characters in each subsequence shown is important.
	Function Description
	Complete the merge_the_tools function in the editor below.
	merge_the_tools has the following parameters:

Code:

```
def merge_the_tools(string, k):
    # your code goes here

    l = len(string)//k
    for i in range(l):
        print(''.join(dict.fromkeys(string[i*k:(i*k)+k])))

if __name__ == '__main__':
    string, k = input(), int(input())
    merge_the_tools(string, k)
```

Test Cases:

✓ Test case 0

✓ Test case 1

✓ Test case 2

✓ Test case 3

✓ Test case 4

✓ Test case 5

✓ Test case 6

Compiler Message

Success

Input (stdin)

Download

1	AABCAAADA
2	3

Expected Output

Download

1	AB
2	CA
3	AD

4. Time Delta

Task and Code:

H

HackerRank

Prepare > Python > Date and Time > Time Delta

Exit Full Screen View

P

Problem

When users post an update on social media, such as a URL, image, status update etc., other users in their network are able to view this new post on their news feed. Users can also see exactly when the post was published, i.e., how many hours, minutes or seconds ago.

Since sometimes posts are published and viewed in different time zones, this can be confusing. You are given two timestamps of one such post that a user can see on his newsfeed in the following format:

Day dd Mon yyyy hh:mm:ss +xxxx

Here +xxxx represents the time zone. Your task is to print the absolute difference (in seconds) between them.

Input Format

The first line contains T , the number of testcases.

Each testcase contains 2 lines, representing time t_1 and time t_2 .

Constraints

- Input contains only valid timestamps
- $year \leq 3000$.

S

Submissions

L

Leaderboard

Change Theme

Language: Python 3

⌂

```
#!/bin/python3
import math
import os
import random
import re
import sys
from datetime import datetime

def time_delta(t1, t2):
    time_format = '%a %d %b %Y %H:%M:%S %z'
    t1 = datetime.strptime(t1, time_format)
    t2 = datetime.strptime(t2, time_format)
    return str(int(abs((t1-t2).total_seconds())))

if __name__ == '__main__':
    fptr = open(os.environ['OUTPUT_PATH'], 'w')
    t = int(input())
    for t_itr in range(t):
        t1 = input()
        t2 = input()
        delta = time_delta(t1, t2)
        fptr.write(delta + '\n')
    fptr.close()
```

Line: 20 Col: 33

Test Cases:

✓ Test case 0

✓ Test case 1

✓ Test case 2

Compiler Message

Success

Input (stdin)

Download

1 2
2 Sun 10 May 2015 13:54:36 -0700
3 Sun 10 May 2015 13:54:36 -0000
4 Sat 02 May 2015 19:54:36 +0530
5 Fri 01 May 2015 13:54:36 -0000

Expected Output

Download

1 25200
2 88200

5. Find Angle MBC

Task and Code:

HackerRank

Prepare > Python > Math > Find Angle MBC

Problem

is a right triangle, 90° at B .
Therefore, $\angle ABC = 90^\circ$.
Point M is the midpoint of hypotenuse AC .
You are given the lengths AB and BC .
Your task is to find $\angle MBC$ (angle θ° , as shown in the figure) in degrees.

Submissions

Leaderboard

Change Theme

Language Python 3

Exit Full Screen View

```
# Enter your code here. Read input from STDIN. Print output to STDOUT
import math

ab=int(input())
bc=int(input())

ca=math.hypot(ab,bc)
mc=ca/2

bca=math.asin(1*ab/ca)
bn=math.sqrt((bc**2+mc**2)-(2*bc*mc*math.cos(bca)))
mbc=math.asin(math.sin(bca)*mc/bn)

print(int(round(math.degrees(mbc),0)),'\u00B0',sep='')

if __name__ == '__main__':
    AB = float(input())
    BC = float(input())
    MBC_rad = math.atan2(AB, BC)
    MBC_deg = round(math.degrees(MBC_rad))
    print(f'{MBC_deg}\u00B0')
```

Line: 13 Col: 1

Test Cases:

✓ Test case 0

✓ Test case 1

✓ Test case 2

✓ Test case 3

✓ Test case 4

✓ Test case 5

Compiler Message

Success

Input (stdin)

Download

1 10
2 10

Expected Output

Download

1 45°

6. No Idea!

Task and Code:

There is an array of n integers. There are also 2 disjoint sets, A and B , each containing m integers. You like all the integers in set A and dislike all the integers in set B . Your initial happiness is 0. For each i integer in the array, if $i \in A$, you add 1 to your happiness. If $i \in B$, you add -1 to your happiness. Otherwise, your happiness does not change. Output your final happiness at the end.

Note: Since A and B are sets, they have no repeated elements. However, the array might contain duplicate elements.

Constraints

- $1 \leq n \leq 10^5$
- $1 \leq m \leq 10^5$
- $1 \leq \text{Any integer in the input} \leq 10^9$

Input Format

The first line contains integers n and m separated by a space.

The second line contains n integers, the elements of the array.

The third and fourth lines contain m integers, A and B , respectively.

```
# Enter your code here. Read input from STDIN. Print output to STDOUT
if __name__ == "__main__":
    happiness = 0
    n, m = map(int, input().strip().split(' '))
    arr = list(map(int, input().strip().split(' ')))

    good = set(map(int, input().strip().split(' ')))
    bad = set(map(int, input().strip().split(' ')))

    for i in arr:
        if i in good:
            happiness += 1
        elif i in bad:
            happiness -= 1

    print(happiness)
```

Test Cases:

Test case 0

Test case 1

Test case 2

Test case 3

Test case 4

Test case 5

Test case 6

Compiler Message

Success

Input (stdin)

1	3 2
2	1 5 3
3	3 1
4	5 7

Expected Output

1	1
---	---

7. Word Order

Task and Code:

Problem

You are given n words. Some words may repeat. For each word, output its number of occurrences. The output order should correspond with the input order of appearance of the word. See the sample input/output for clarification.

Note: Each input line ends with a "\n" character.

Constraints:

- $1 \leq n \leq 10^6$
- The sum of the lengths of all the words do not exceed 10^6
- All the words are composed of lowercase English letters only.

Input Format

The first line contains the integer, n .

The next n lines each contain a word.

Output Format

Output 2 lines.

On the first line, output the number of distinct words from the input.

On the second line, output the number of occurrences for each distinct word according to their appearance in the input.

```
# Enter your code here. Read input from STDIN. Print output to STDOUT
n=int(input())

words=[input() for i in range(n)]

words_occurrences={}

for word in words:
    words_occurrences[word]=0

for word in words:
    words_occurrences[word]+=1

print(len(words_occurrences))

occurrences=words_occurrences.values()

for i in occurrences:
    print(i, end=" ")

Line: 20 Col: 1
```

Test Cases:

✓ Test case 0	Success
✓ Test case 1	
✓ Test case 2	
✓ Test case 3	
✓ Test case 4	
✓ Test case 5	
✓ Test case 6	

Input (stdin)		Download
1	4	
2	bcdef	
3	abcdefg	
4	bcde	
5	bcdef	

Expected Output		Download
1	3	
2	2 1 1	

8. Compress the String!

Task and Code:

HackerRank

Prepare > Python > Itertools > Compress the String!

Exit Full Screen View

Problem

Submissions

Leaderboard

In this task, we would like for you to appreciate the usefulness of the `groupby()` function of `itertools`. To read more about this function, [Check this out](#).

You are given a string S . Suppose a character 'c' occurs consecutively X times in the string. Replace these consecutive occurrences of the character 'c' with (X, c) in the string.

For a better understanding of the problem, check the explanation.

Input Format

A single line of input consisting of the string S .

Output Format

A single line of output consisting of the modified string.

Constraints

All the characters of S denote integers between 0 and 9.

$1 \leq |S| \leq 10^4$

Change Theme

Language Python 3

Enter your code here. Read input from STDIN. Print output to STDOUT

from itertools import groupby

def compress_string(s):

compressed = []

for key, group in groupby(s):

count = len(list(group))

compressed.append(f"({count}, {key})")

return ' '.join(compressed)

if __name__ == "__main__":

s = input()

result = compress_string(s)

print(result)

Test Cases:

✓ Test case 0	Compiler Message
✓ Test case 1	Success
✓ Test case 2	
✓ Test case 3	
✓ Test case 4	
✓ Test case 5	

Input (stdin)		Download
1	1222311	

Expected Output		Download
1	(1, 1) (3, 2) (1, 3) (2, 1)	

9. Company Logo

Task and Code:

HackerRank | Prepare > Python > Collections > Company Logo

Problem

A newly opened multinational brand has decided to base their company logo on the three most common characters in the company name. They are now trying out various combinations of company names and logos based on this condition. Given a string s , which is the company name in lowercase letters, your task is to find the top three most common characters in the string.

- Print the three most common characters along with their occurrence count.
- Sort in descending order of occurrence count.
- If the occurrence count is the same, sort the characters in alphabetical order.

For example, according to the conditions described above, `GOOGLE` would have it's logo with the letters `G, O, L`.

Input Format

A single line of input containing the string S .

```
#!/bin/python3
import math
import os
import random
import re
import sys

7
if __name__ == '__main__':
    s = input()
    char_freq = {}

    for char in s:
        char_freq[char] = char_freq.get(char, 0) + 1

    sorted_chars = sorted(char_freq.items(), key=lambda x: (-x[1], x[0]))

    for char, count in sorted_chars[:3]:
        print(f'{char} {count}')
```

Test Cases:

✓ **Test case 0**

✓ Test case 1

✓ Test case 2

✓ Test case 3

✓ Test case 4

✓ Test case 5

Compiler Message

Success

Input (stdin) [Download](#)

```
1 aabbbccde
```

Expected Output [Download](#)

```
1 b 3
2 a 2
3 c 2
```

10. Piling Up!

Task:

Problem

There is a horizontal row of n cubes. The length of each cube is given. You need to create a new vertical pile of cubes. The new pile should follow these directions: if $cube[i]$ is on top of $cube[j]$ then $sideLength[j] \geq sideLength[i]$.

When stacking the cubes, you can only pick up either the leftmost or the rightmost cube each time. Print Yes if it is possible to stack the cubes. Otherwise, print No.

Code:

```
1 def can_stack_cubes(n, cubes):
2     left_index = 0
3     right_index = n - 1
4     top_cube = float('inf')
5
6     while left_index <= right_index:
7         if cubes[left_index] >= cubes[right_index] and cubes
8         [left_index] <= top_cube:
9             top_cube = cubes[left_index]
10            left_index += 1
11
12        elif cubes[right_index] >= cubes[left_index] and cubes
13        [right_index] <= top_cube:
14            top_cube = cubes[right_index]
15            right_index -= 1
16
17        else:
18            return "No"
19
20    return "Yes"
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
```

Test Cases:

✓ Test case 0

✓ Test case 1

✓ Test case 2

✓ Test case 3

✓ Test case 4

Compiler Message

Success

Input (stdin)

1 2

2 6

3 4 3 2 1 3 4

4 3

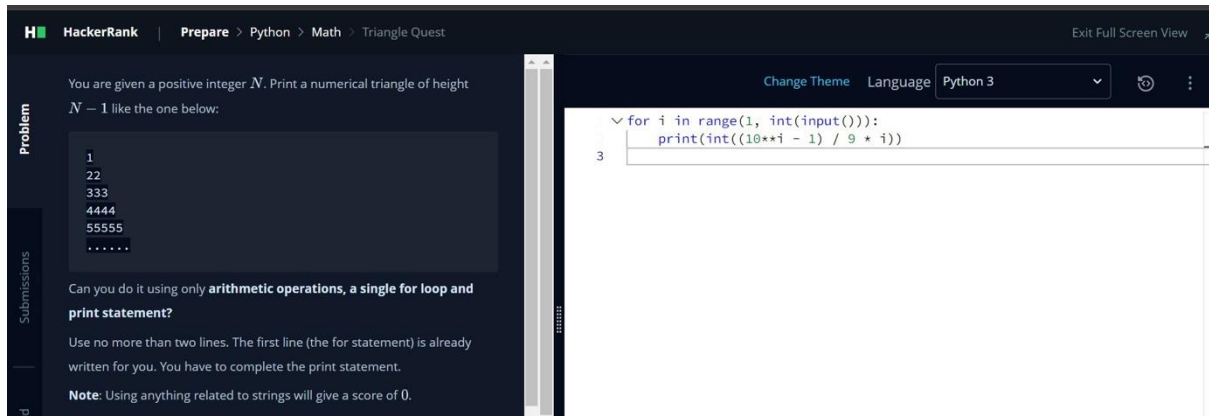
5 1 3 2

Expected Output

1 Yes

11. Triangle Quest

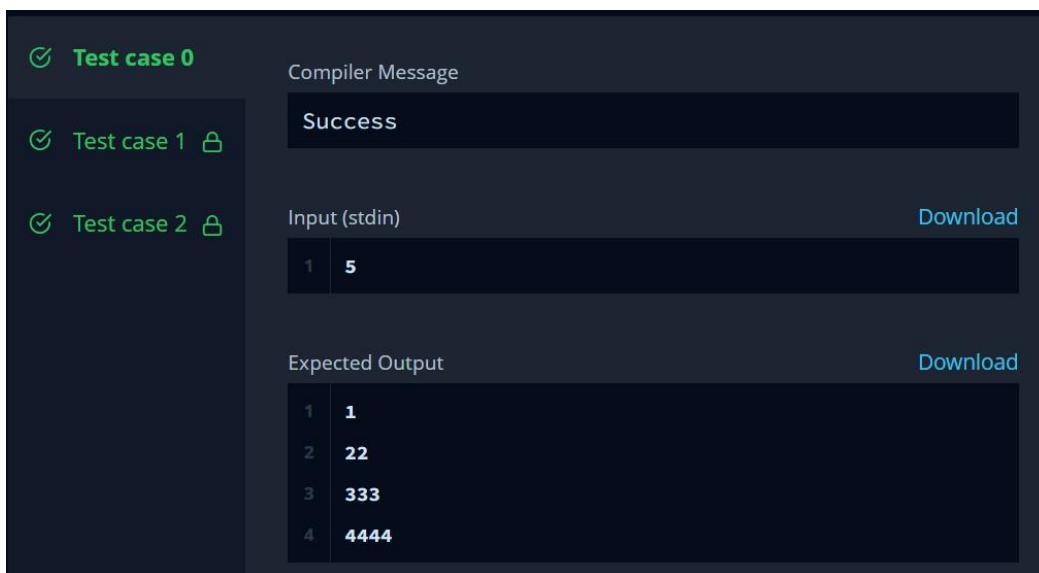
Task and Code:



The screenshot shows the HackerRank interface for the 'Triangle Quest' problem. The problem description states: 'You are given a positive integer N . Print a numerical triangle of height $N - 1$ like the one below:'. An example triangle for $N=6$ is shown: 1, 22, 333, 4444, 55555, 666666. The instructions specify using only arithmetic operations, a single for loop, and a print statement, with a limit of two lines of code. The code editor on the right contains the following Python 3 code:

```
for i in range(1, int(input())):
    print(int((10**i - 1) / 9 * i))
```

Test Cases:

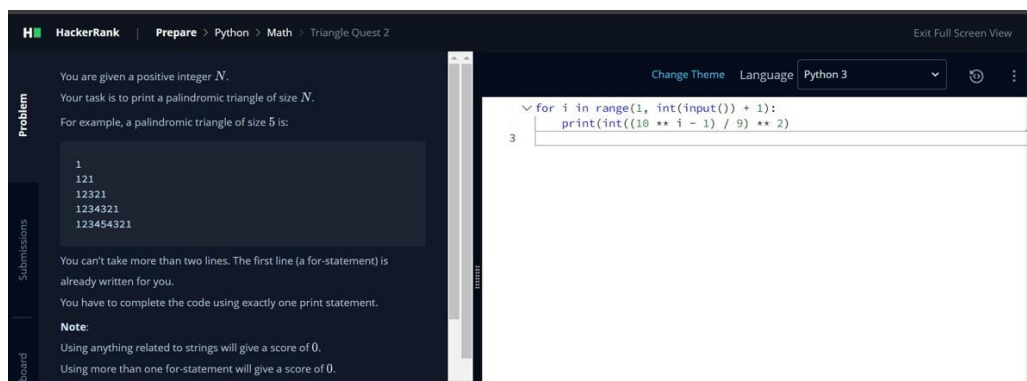


The screenshot displays the test cases for the 'Triangle Quest' problem. On the left, three test cases are listed, all marked as successful. The 'Compiler Message' section shows 'Success'. The 'Input (stdin)' section shows the input '5'. The 'Expected Output' section shows the following output:

```
1 1
2 22
3 333
4 4444
```

12. Triangle Quest 2

Task and Code:



The screenshot shows the HackerRank interface for the 'Triangle Quest 2' problem. The problem description states: 'You are given a positive integer N . Your task is to print a palindromic triangle of size N . For example, a palindromic triangle of size 5 is:'. An example triangle for $N=5$ is shown: 1, 121, 12321, 1234321, 123454321. The instructions specify using exactly one print statement. The code editor on the right contains the following Python 3 code:

```
for i in range(1, int(input()) + 1):
    print(int((10 ** i - 1) / 9) ** 2)
```

Test Cases:

Test case 0 ✓

Test case 1 ✓

Test case 2 ✓

Test case 3 ✓

Test case 4 ✓

Test case 5 ✓

Compiler Message

Success

Input (stdin)

1 5

Expected Output

1 1

2 121

3 12321

4 1234321

5 123454321

13. Iterables and Iterators

Task and Code:

HackerRank | Prepare > Python > Itertools > Iterables and Iterators

Problem

The itertools module standardizes a core set of fast, memory efficient tools that are useful by themselves or in combination. Together, they form an iterator algebra making it possible to construct specialized tools succinctly and efficiently in pure Python.

To read more about the functions in this module, check out their [documentation here](#).

You are given a list of N lowercase English letters. For a given integer K , you can select any K indices (assume 1-based indexing) with a uniform probability from the list.

Find the probability that at least one of the K indices selected will contain the letter: 'a'.

Input Format

The input consists of three lines. The first line contains the integer N , denoting the length of the list. The next line consists of N space-separated lowercase English letters, denoting the elements of the list.

The third and the last line of input contains the integer K , denoting the number of indices to be selected.

Submissions

Leaderboard

Change Theme Language Python 3

```
from itertools import combinations

def probability_of_a(N, letters, K):
    total_combinations = list(combinations(range(1, N+1), K))
    combinations_without_a = [comb for comb in total_combinations if
                              'a' not in [letters[i-1] for i in comb]]

    probability_without_a = len(combinations_without_a) / len
    (total_combinations)
    probability_with_a = 1 - probability_without_a

    return round(probability_with_a, 4)

if __name__ == "__main__":
    N = int(input())
    letters = input().split()
    K = int(input())

    result = probability_of_a(N, letters, K)
    print(result)
```

19

Line: 19 Col: 1

Test Cases:

Test case 0 ✓

Test case 1 ✓

Test case 2 ✓

Test case 3 ✓

Test case 4 ✓

Test case 5 ✓

Compiler Message

Success

Input (stdin)

1 4

2 a a c d

3 2

Expected Output

1 0.833333333333

14. Classes: Dealing With Complex Numbers

Task:

Problem	<p>For this challenge, you are given two complex numbers, and you have to print the result of their addition, subtraction, multiplication, division and modulus operations.</p> <p>The real and imaginary precision part should be correct up to two decimal places.</p>
Submissions	<p>Input Format</p> <p>One line of input: The real and imaginary part of a number separated by a space.</p> <p>Output Format</p> <p>For two complex numbers C and D, the output should be in the following sequence on separate lines:</p> <ul style="list-style-type: none">• $C + D$• $C - D$• $C * D$• C / D• $\text{mod}(C)$• $\text{mod}(D)$
Leaderboard	

Code:

```
1 import math
2
3 class Complex(object):
4     def __init__(self, real, imaginary):
5         self.real = real
6         self.imaginary = imaginary
7
8     def __add__(self, no):
9         return Complex(self.real + no.real, self.imaginary + no.imaginary)
10
11     def __sub__(self, no):
12         return Complex(self.real - no.real, self.imaginary - no.imaginary)
13
14     def __mul__(self, no):
15         return Complex(self.real * no.real - self.imaginary * no.
16             imaginary, self.real * no.imaginary + self.imaginary * no.real)
17
18     def __truediv__(self, no):
19         denominator = no.real**2 + no.imaginary**2
20         real_part = (self.real * no.real + self.imaginary * no.imaginary)
21         / denominator
```

```

20         imag_part = (self.imaginary * no.real - self.real * no.imaginary)
21     / denominator
22     return Complex(real_part, imag_part)
23
24     def mod(self):
25         return Complex(math.sqrt(self.real**2 + self.imaginary**2), 0)
26
27     def __str__(self):
28         if self.imaginary == 0:
29             result = "%.2f+0.00i" % (self.real)
30         elif self.real == 0:
31             if self.imaginary >= 0:
32                 result = "0.00+%.2fi" % (self.imaginary)
33             else:
34                 result = "0.00-%.2fi" % (abs(self.imaginary))
35         elif self.imaginary > 0:
36             result = "%.2f+%.2fi" % (self.real, self.imaginary)
37         else:
38             result = "%.2f-%.2fi" % (self.real, abs(self.imaginary))
39         return result

```

```

37
38
39
40 if __name__ == '__main__':
41     c = map(float, input().split())
42     d = map(float, input().split())
43     x = Complex(*c)
44     y = Complex(*d)
45     print(*map(str, [x+y, x-y, x*y, x/y, x.mod(), y.mod()]), sep='\n')

```

Test Cases:

Test Case	Input (stdin)	Expected Output
Test case 0	2 1	7.00+7.00i
Test case 1	5 6	-3.00-5.00i
Test case 2		4.00+17.00i
Test case 3		0.26-0.11i
Test case 4		2.24+0.00i
Test case 5		7.81+0.00i
Test case 6		

15. Athlete Sort

Task and Code:

HackerRank

Prepare > Python > Built-ins > Athlete Sort

Exit Full Screen View

Change Theme Language Python 3

Problem

You are given a spreadsheet that contains a list of N athletes and their details (such as age, height, weight and so on). You are required to sort the data based on the K^{th} attribute and print the final resulting table. Follow the example given below for better understanding.

Rank	Age	Height (in cm)	Rank	Age	Height (in cm)
1	32	190	5	24	176
2	35	175	4	26	195
3	41	188	1	32	190
4	26	195	2	35	175
5	24	176	3	41	188

Note that K is indexed from 0 to $M - 1$, where M is the number of attributes.

Note: If two attributes are the same for different rows, for example, if two athletes are of the same age, print the row that appeared first in the input.

Submissions

Leaderboard

```
#!/bin/python3
import math
import os
import random
import re
import sys

if __name__ == '__main__':
    nm = input().split()
    n = int(nm[0])
    m = int(nm[1])
    arr = []

    for _ in range(n):
        arr.append(list(map(int, input().rstrip().split())))

    k = int(input())
    arr.sort(key=lambda x: x[k])

    for row in arr:
        print(" ".join(map(str, row)))
```

Test Cases:

Test case 0

Test case 1

```
1 5 3
2 10 2 5
3 7 1 0
4 9 9 9
5 1 23 12
6 6 5 9
7 1
```

Expected Output

Download

```
1 7 1 0
2 10 2 5
3 6 5 9
4 9 9 9
```

16. ginortS

Task and Code:

HackerRank


Prepare > Python > Built-ins > ginortS

Exit Full Screen View

Change Theme Language Python 3

Problem

You are given a string S . S contains alphanumeric characters only.



Your task is to sort the string S in the following manner:

- All sorted lowercase letters are ahead of uppercase letters.
- All sorted uppercase letters are ahead of digits.
- All sorted odd digits are ahead of sorted even digits.

Input Format

A single line of input contains the string S .

Constraints

- $0 < \text{len}(S) < 1000$

Output Format

Output the sorted string S .

Submissions

Leaderboard

```
# Enter your code here. Read input from STDIN. Print output to STDOUT
def sort(s):
    sorted_string = sorted(s, key=lambda x: (x.isdigit(), x.isdigit() and int(x) % 2 == 0, x.isupper(), x.islower(), x))
    result = ''.join(sorted_string)
    return result

if __name__ == "__main__":
    S = input()
    result = sort(S)
    print(result)
```

Line: 11 Col: 1

Test Cases:

✓ Test case 0	Compiler Message
✓ Test case 1	Success
✓ Test case 2	Input (stdin) Download
✓ Test case 3	1 <code>Sorting1234</code>
✓ Test case 4	Expected Output Download
	1 <code>ginortS1324</code>

17. Validating Email Addresses With a Filter

Task:

You are given an integer N followed by N email addresses. Your task is to print a list containing only valid email addresses in lexicographical order.

Valid email addresses must follow these rules:

- It must have the username@websiteName.extension format type.
- The username can only contain letters, digits, dashes and underscores $[a - z], [A - Z], [0 - 9], [-]$.
- The website name can only have letters and digits $[a - z], [A - Z], [0 - 9]$.
- The extension can only contain letters $[a - z], [A - Z]$.
- The maximum length of the extension is 3.

Concept

A filter takes a function returning True or False and applies it to a sequence, returning a list of only those members of the sequence where the function returned True. A Lambda function can be used with filters.

Let's say you have to make a list of the squares of integers from 0 to 9 (both included).


```
>> l = list(range(10))
>> l = list(map(lambda x:x*x, l))
```

Now, you only require those elements that are greater than 10 but less than 80.

```
>> l = list(filter(lambda x: x > 10 and x < 80, l))
```

Easy, isn't it?

Example

Complete the function fun in the editor below.

fun has the following paramters:

- string s: the string to test

Returns

- boolean: whether the string is a valid email or not

Code:

```
1  def fun(s):
2
3      if '@' not in s:
4          return False
5
6      parts = s.split('@')
7
8      if len(parts) != 2:
9          return False
10
11     username, rest = parts
12
13     if not username.replace('_', '').replace('-', '').isalnum():
14         return False
15
16     if '.' not in rest:
17         return False
18
19     website, extension = rest.split('.')
20
21     if not (website.isalnum() and extension.isalpha() and len
22             (extension) <= 3):
23         return False
24     return True
25
26 def filter_mail(emails):
27     return list(filter(fun, emails))
28
29 if __name__ == '__main__':
30     n = int(input())
31     emails = []
32     for _ in range(n):
33         emails.append(input())
34
35 filtered_emails = filter_mail(emails)
36 filtered_emails.sort()
37 print(filtered_emails)
```

Test Cases:

Test case 0 ✓

Test case 1 ✓

Test case 2 ✓

Test case 3 ✓

Test case 4 ✓

Test case 5 ✓

Test case 6 ✓

Compiler Message

Success

Input (stdin) [Download](#)

```
1 3
2 lara@hackerrank.com
3 brian-23@hackerrank.com
4 britts_54@hackerrank.com
```

Expected Output [Download](#)

```
1 ['brian-23@hackerrank.com',
  'britts_54@hackerrank.com', 'lara@hackerrank.com']
```

18. Reduce Function

Task and Code:

Problem

Given a list of rational numbers, find their product.

Concept

The `reduce()` function applies a function of two arguments cumulatively on a list of objects in succession from left to right to reduce it to one value. Say you have a list, say `[1,2,3]` and you have to find its sum.

```
>>> reduce(lambda x, y : x + y, [1,2,3])
6
```

You can also define an initial value. If it is specified, the function will assume initial value as the value given, and then reduce. It is equivalent to adding the initial value at the beginning of the list. For example:

```
>>> reduce(lambda x, y : x + y, [1,2,3], -3)
3
```

```
>>> from fractions import gcd
>>> reduce(gcd, [2,4,8], 3)
1
```

Submissions

Leaderboard

Code

```
> from fractions import Fraction...
3
def product(fracs):
    t = Fraction(reduce(lambda x, y: x * y, fracs))
    return t.numerator, t.denominator

if __name__ == '__main__':
    fracs = []
    for _ in range(int(input())):
        fracs.append(Fraction(*map(int, input().split())))
    result = product(fracs)
    print(*result)
```

Line: 3 Col: 1

Test Cases:

Test case 0 ✓

Test case 1 ✓

Test case 2 ✓

Test case 3 ✓

Test case 4 ✓

Test case 5 ✓

Compiler Message

Success

Input (stdin) [Download](#)

```
1 21
2 684025282 932952183
3 349232934 278093065
4 778706161 742081687
5 374870211 874099626
6 849763633 211127281
```

19. Regex Substitution

Task and Code:

The `re.sub()` tool (sub stands for substitution) evaluates a pattern and, for each valid match, it calls a method (or lambda).

The method is called for all matches and can be used to modify strings in different ways.

The `re.sub()` method returns the modified string as an output.

Learn more about `re.sub()`.

Transformation of Strings

Code

```
import re

#Squaring numbers
def square(match):
    number = int(match.group(0))
    return str(number**2)

print re.sub(r"[d+]", square, "1 2 3 4 5 6 7 8 9")
```

Output

```
1 4 9 16 25 36 49 64 81
```

Change Theme Language Python 3

```
import re

for i in range(int(input())):
    s = re.sub("(?<=\\s)&&(?=\\s)", "and", input())
    print(re.sub("(?<=\\s)\\|\\|(?=\\s)", "or", s))
```

Line: 4 Col: 1

☐ Upload Code as File ☐ Test against custom input

Test Cases:

✓ Test case 0

✓ Test case 1

✓ Test case 2

✓ Test case 3

✓ Test case 4

✓ Test case 5

✓ Test case 6

Expected Output

```
1 a = 1;
2 b = input();
3
4 if a + b > 0 and a - b < 0:
5     start()
6 elif a*b > 10 or a/b < 1:
7     stop()
8 print set(list(a)) | set(list(b))
9 #Note do not change && or ||| or & or |
10 #Only change those '&&' which have space on both
11 #Only change those '||' which have space on both
    sides.
```

Download

20. Validating Credit Card Numbers

Task:

Problem

You and Fredrick are good friends. Yesterday, Fredrick received N credit cards from **ABCD Bank**. He wants to verify whether his credit card numbers are valid or not. You happen to be great at regex so he is asking for your help!

A valid credit card from **ABCD Bank** has the following characteristics:

Submissions

- ▶ It must start with a 4, 5 or 6.
- ▶ It must contain exactly 16 digits.
- ▶ It must only consist of digits (0-9).
- ▶ It may have digits in groups of 4, separated by one hyphen "-".
- ▶ It must **NOT** use any other separator like ' ', '_', etc.
- ▶ It must **NOT** have 4 or more consecutive repeated digits.

Code:

```
1 def is_valid(card_number):
2     if not (16 == len(card_number) or (19 == len(card_number) and
3         card_number[4] == '-' and card_number[9] == '-' and card_number[14] ==
4         '-')):
5         return "Invalid"
6
7     if card_number[0] not in '456':
8         return "Invalid"
9
10    card_number = card_number.replace('-', '')
11
12    if not card_number.isdigit():
13        return "Invalid"
14
15    for i in range(len(card_number) - 3):
16        if card_number[i] == card_number[i + 1] == card_number[i + 2]
17        == card_number[i + 3]:
18            return "Invalid"
19
20    return "Valid"
21
22 N = int(input())
23
24 for _ in range(N):
25     card_number = input()
26
27     result = is_valid(card_number)
28     if result == "Valid":
29         print("Valid")
30     else:
31         print("Invalid")
32
33
```

Test Cases:

✓ Test case 0

✓ Test case 1

✓ Test case 2

✓ Test case 3

✓ Test case 4

✓ Test case 5

3 5123-4567-8912-3456

4 61234-567-8912-3456

5 4123356789123456

6 5133-3367-8912-3456

7 5123 - 3567 - 8912 - 3456

Expected Output

Download

1 Valid

2 Valid

3 Invalid

4 Valid

5 Invalid

6 Invalid

21. Words Score

Task and Code:

Problem

In this challenge, the task is to debug the existing code to successfully execute all provided test files.

Consider that vowels in the alphabet are a, e, i, o, u and y.

Function `score_words` takes a list of lowercase words as an argument and returns a score as follows:

The score of a single word is 2 if the word contains an even number of vowels. Otherwise, the score of this word is 1. The score for the whole list of words is the sum of scores of all words in the list.

Debug the given function `score_words` such that it returns a correct score.

Your function will be tested on several cases by the locked template code.

Input Format

The input is read by the provided locked code template. In the first line, there is a single integer `n` denoting the number of words. In the second line, there are `n` space-separated lowercase words.

```
def is_vowel(letter):
    return letter in ['a', 'e', 'i', 'o', 'u', 'y']

def score_words(words):
    score = 0
    for word in words:
        num_vowels = 0
        for letter in word:
            if is_vowel(letter):
                num_vowels += 1
            if num_vowels % 2 == 0:
                score += 2
            else:
                score += 1
    return score

n = int(input())
words = input().split()
print(score_words(words))
```

Test Cases:

Test case 0

Test case 1

Test case 2

Test case 3

Test case 4

Test case 5

Test case 6

Compiler Message

Success

Input (stdin)

Download

1	2
2	hacker book

Expected Output

Download

1	4
---	---

22. Default Arguments

Task:

In this challenge, the task is to debug the existing code to successfully execute all provided test files.

Python supports a useful concept of default argument values. For each keyword argument of a function, we can assign a default value which is going to be used as the value of said argument if the function is called without it. For example, consider the following increment function:

```
def increment_by(n, increment=1):  
    return n + increment
```

The function works like this:

```
>>> increment_by(5, 2)  
7  
>>> increment_by(4)  
5  
>>>
```

Debug the given function `print_from_stream` using the default value of one of its arguments.

The function has the following signature:

```
def print_from_stream(n, stream)
```

This function should print the first n values returned by `get_next()` method of `stream` object provided as an argument. Each of these values should be printed in a separate line.

Whenever the function is called without the `stream` argument, it should use an instance of `EvenStream` class defined in the code stubs below as the value of `stream`.

Your function will be tested on several cases by the locked template code.

Input Format

The input is read by the provided locked code template. In the first line, there is a single integer q denoting the number of queries. Each of the following q lines contains a `stream_name` followed by integer n , and it corresponds to a single test for your function.

Code:

Hard Level Tasks:

23. Maximize It!

Task and Code:

HackerRank | Prepare > Python > Itertools > Maximize It!

Problem

You are given a function $f(X) = X^2$. You are also given K lists. The i^{th} list consists of N_i elements.

You have to pick one element from each list so that the value from the equation below is maximized:

$$S = (f(X_1) + f(X_2) + \dots + f(X_K)) \% M$$

X_i denotes the element picked from the i^{th} list. Find the maximized value S_{max} obtained.

$\%$ denotes the modulo operator.

Note that you need to take exactly one element from each list, not necessarily the largest element. You add the squares of the chosen elements and perform the modulo operation. The maximum value that you can obtain, will be the answer to the problem.

Input Format

The first line contains 2 space separated integers K and M .

The next K lines each contains an integer N_i , denoting the number of elements in the i^{th} list, followed by N_i space separated integers denoting the elements in the list.

Submissions

Leaderboard

Code Editor

```
from itertools import product

def maximize_S(K, M, lists):
    max_S = 0
    all_combinations = product(*lists)

    for combination in all_combinations:
        current_S = sum(x ** 2 for x in combination) % M
        max_S = max(max_S, current_S)

    return max_S

if __name__ == "__main__":
    K, M = map(int, input().split())
    lists = [list(map(int, input().split()[1:])) for _ in range(K)]

    result = maximize_S(K, M, lists)
    print(result)
```

Line: 19 Col: 1

Test Cases:

Test case 0

Test case 1

Test case 2

Test case 3

Test case 4

Test case 5

Test case 6

Compiler Message

Success

Input (stdin) [Download](#)

```
1 7 867
2 7 6429964 4173738 9941618 2744666 5392018 5813128 9452095
3 7 6517823 4135421 6418713 9924958 9370532 7940650 2027017
4 7 1506500 3460933 1550284 3679489 4538773 5216621 5645660
5 7 7443563 5181142 8804416 8726696 5358847 7155276 4433125
6 7 2230555 3920370 7851992 1176871 610460 309961
```

24. Validating Postal Codes

Task and Code:

HackerRank | Prepare > Python > Regex and Parsing | Validating Postal Codes

Exit Full Screen View

Problem

A valid postal code P have to fulfill both below requirements:

1. P must be a number in the range from 100000 to 999999 inclusive.
2. P must not contain more than one alternating repetitive digit pair.

Alternating repetitive digits are digits which repeat immediately after the next digit. In other words, an alternating repetitive digit pair is formed by two equal digits that have just a single digit between them.

For example:

```
121426 # Here, 1 is an alternating repetitive digit.
523563 # Here, NO digit is an alternating repetitive dig
552523 # Here, both 2 and 5 are alternating repetitive d
```

Your task is to provide two regular expressions `regex_integer_in_range` and `regex_alternating_repetitive_digit_pair`. Where:

- `regex_integer_in_range` should match only integers range from 100000 to 999999 inclusive
- `regex_alternating_repetitive_digit_pair` should find alternating repetitive digits pairs in a given string.

Both these regular expressions will be used by the provided code template

Submissions

Leaderboard

Code Editor

Change Theme | Language: Python 3

```
regex_integer_in_range = r"([1-9][0-9]{5})"
regex_alternating_repetitive_digit_pair = r"(?=(\d)\d\d)"

import re
P = input()

7 print(bool(re.match(regex_integer_in_range, P))
and len(re.findall(regex_alternating_repetitive_digit_pair, P)) < 2)
```

Line: 7 Col: 6

Upload Code as File | Test against custom input | Run Code | Submit Code

Test Cases:

✓ **Test case 0**

✓ Test case 1

✓ Test case 2

✓ Test case 3

✓ Test case 4

Compiler Message

Success

Input (stdin) [Download](#)

```
1 110000
```

Expected Output [Download](#)

```
1 False
```

25. Matrix Script

Task:

Neo has a complex matrix script. The matrix script is a $N \times M$ grid of strings. It consists of alphanumeric characters, spaces and symbols (!, @, #, \$, %, &).

Matrix Script

T	s	i
h	%	x
i		#
s	M	
\$	a	
#	t	%
i	r	!

Matrix Decoded

```
This$#is% Matrix# %!
```

To decode the script, Neo needs to read each column and select only the alphanumeric characters and connect them. Neo reads the column from top to bottom and starts reading from the leftmost column.

If there are symbols or spaces between two alphanumeric characters of the decoded script, then Neo replaces them with a single space ' ' for better readability.

Neo feels that there is no need to use 'if' conditions for decoding.

Alphanumeric characters consist of: [A-Z, a-z, and 0-9].

Code:

```
1  import math
2  import os
3  import random
4  import re
5  import sys
6  first_multiple_input = input().rstrip().split()
7  n = int(first_multiple_input[0])
8  m = int(first_multiple_input[1])
9  matrix = []
10 t = []
11 for i in range(n):
12     matrix_item = [x for x in input()]
13     matrix.append(matrix_item)
14 for i in range(m):
15     for j in range(n):
16         t.append(matrix[j][i])
17 s = ''.join(t)
18 path = re.compile(r'\b[ !@#$%&]+\b', re.M)
19 k = re.sub(path, ' ', s)
20 print(k)
21
```

Test Cases:

The screenshot shows a coding challenge interface. On the left, there is a list of test cases, all marked as passed (green checkmark). The main area displays the input for 'Test case 0' and the expected output.

Test case 0 (passed)

Input (stdin)

```
1 7 3
2 Tsi
3 h%x
4 i #
5 sM
6 $a
7 #t%
8 ir!
```

Expected Output

```
1 This is Matrix# %!
```

Download buttons are available for both the input and expected output sections.

