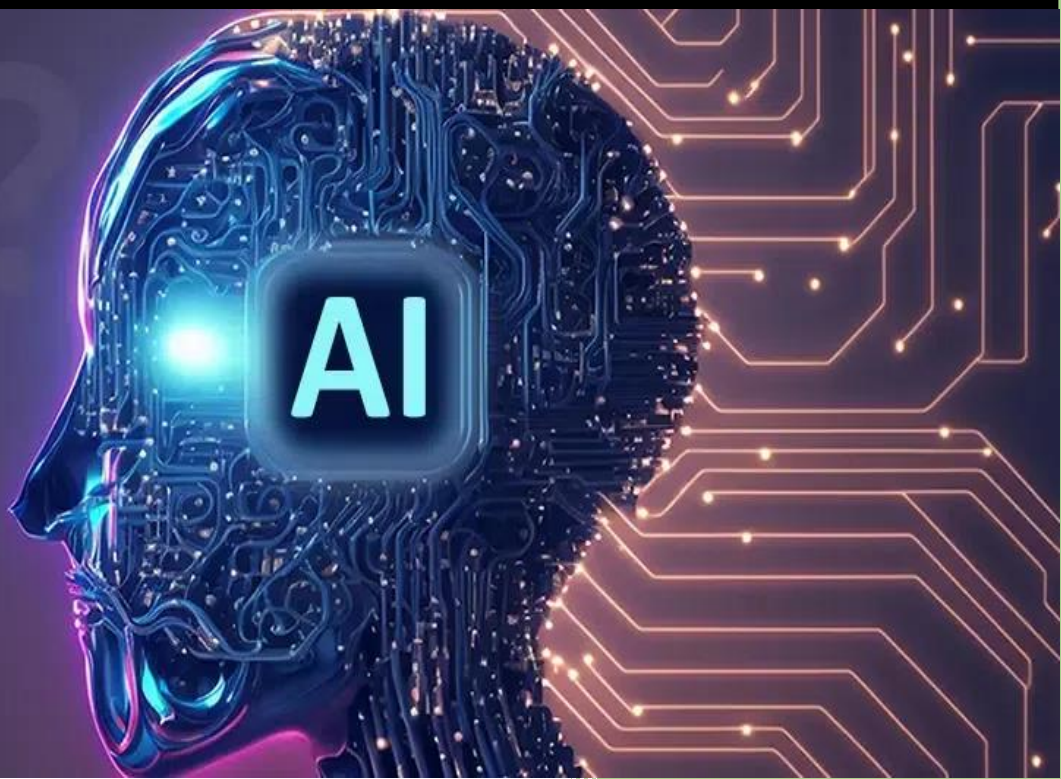


AI Assignment 03



MUHAMMAD ABDULLAH YOUSAF
450767

MEDIUM 01

```
def is_leap(year):  
    leap = False  
    leap = (year % 400 == 0) or (year % 4 == 0 and year % 100 != 0)  
    return leap  
  
year = int(input())  
print(is_leap(year))
```

The screenshot shows the HackerRank interface for the 'Write a function' challenge. The left sidebar contains navigation links: Problem, Submissions, Leaderboard, Discussions, and Tutorial. The main content area on the left displays the problem description for 'is_leap', which asks to determine if a given year is a leap year based on specific conditions. The right panel shows the code editor with the provided Python solution. Below the editor, a green 'Congratulations' banner indicates the challenge was solved. The 'Test Cases' section shows five test cases, all of which passed. The 'Compiler Message' section displays 'Success'. The 'Input (stdin)' and 'Expected Output' sections show the input '2000' and the output 'True' respectively. A 'Hidden Test Case' section is also visible at the bottom.

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.

Input Format

Read `year`, the year to test.

Constraints

$1900 \leq \text{year} \leq 10^5$

Output Format

The function must return a Boolean value (`True/False`). Output is handled by the provided code stub.

Sample Input 0

1990

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)

2000

Expected Output

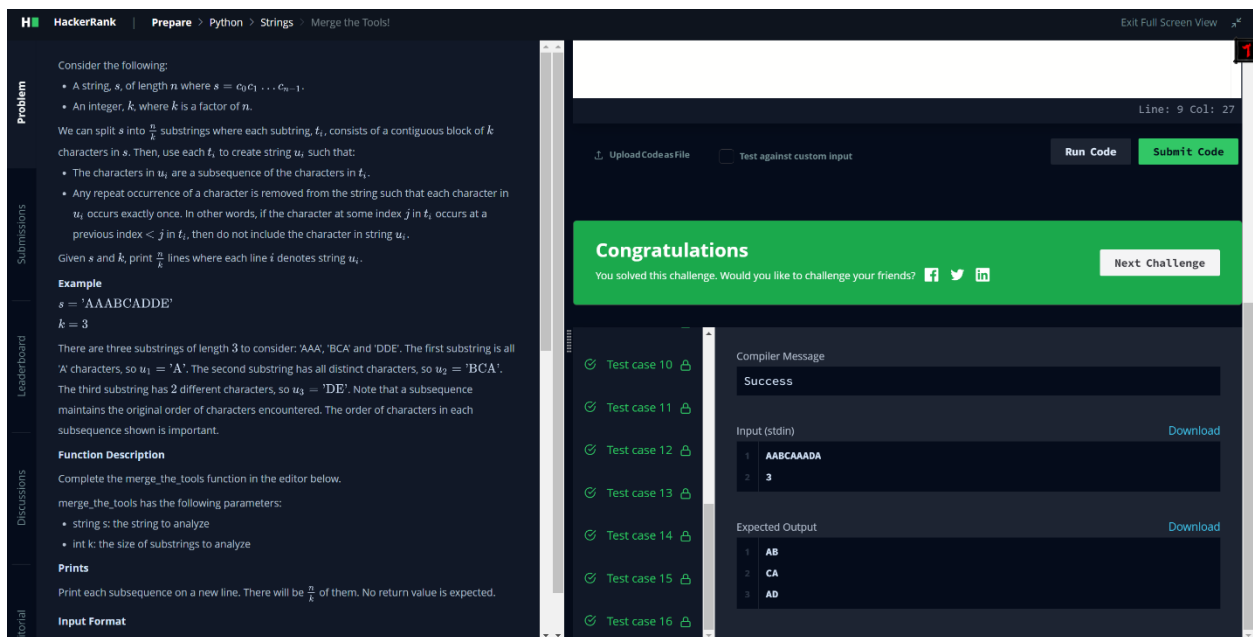
True

Hidden Test Case

MEDIUM 02

```
from collections import OrderedDict

def merge_the_tools(string, k):
    # your code goes here
    for i in range(0, len(string)-k+1, k):
        print(''.join(OrderedDict.fromkeys(string[i:i + k])))
if __name__ == '__main__':
```



HackerRank | Prepare > Python > Strings > Merge the Tools! Exit Full Screen View

Problem

Consider the following:

- 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:

- 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 .

Given s and k , print $\frac{n}{k}$ lines where each line i denotes string u_i .

Example

$s = \text{'AABCCADDE'}$
 $k = 3$

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:

- string s : the string to analyze
- int k : the size of substrings to analyze

Prints

Print each subsequence on a new line. There will be $\frac{n}{k}$ of them. No return value is expected.

Input Format

Line 9 Col: 27

Upload Code as File ☐ Test against custom Input Run Code Submit Code

Congratulations
You solved this challenge. Would you like to challenge your friends? [f](#) [t](#) [in](#) Next Challenge

Test case 10

Test case 11

Test case 12

Test case 13

Test case 14

Test case 15

Test case 16

Compiler Message
Success

Input (stdin) Download

```
1 AABCAAADA
2 3
```

Expected Output Download

```
1 AB
2 CA
3 AD
```

MEDIUM 03

```
import datetime
cas = int(input())
time_format = "%a %d %b %Y %H:%M:%S %z"
for _ in range(cas):
    timestamp1 = input().strip()
    timestamp2 = input().strip()
    time_second1 = datetime.datetime.strptime(timestamp1, time_format)
    time_second2 = datetime.datetime.strptime(timestamp2, time_format)
    print(int(abs((time_second1 - time_second2).total_seconds())))
```

HackerRank | Prepare > Python > Date and Time > Time Delta

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 ≤ 3000 .

Output Format

Print the absolute difference ($t_1 - t_2$) in seconds.

Sample Input 0

```
2
Sun 10 May 2015 13:54:36 -0700
Sun 10 May 2015 13:54:36 -0000
Sat 02 May 2015 19:54:36 +0530
Fri 01 May 2015 13:54:36 -0000
```

Sample Output 0

```
25200
```

Congratulations

You solved this challenge. Would you like to challenge your friends? [f](#) [t](#) [in](#)

Test case 0 Success

Input (stdin)

```
2
Sun 10 May 2015 13:54:36 -0700
Sun 10 May 2015 13:54:36 -0000
Sat 02 May 2015 19:54:36 +0530
Fri 01 May 2015 13:54:36 -0000
```

Expected Output

```
25200
```

MEDIUM 04

```
import math

ab = float(input())
bc = float(input())
ac = math.sqrt(ab**2 + bc**2)
bm = ac / 2.0
mc = bm
# let,
b = mc
c = bm
a = bc
angel_b_radian = math.acos(a / (2 * b))
angel_b_degree = int(round((180 * angel_b_radian) / math.pi))
print(f"{angel_b_degree}{chr(176)}")
```

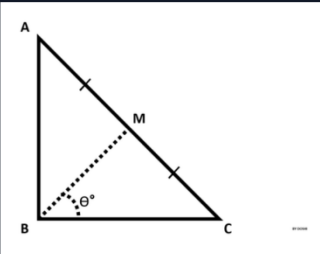
Problem

Submissions

Leaderboard

Discussions

Tutorial



ABC 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.

Input Format

The first line contains the length of side AB .
The second line contains the length of side BC .

Constraints

- $0 < AB \leq 100$
- $0 < BC \leq 100$
- Lengths AB and BC are natural numbers.

Output Format

Exit Full Screen View

Line: 2 Col: 1

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

Congratulations

You solved this challenge. Would you like to challenge your friends? [f](#) [t](#) [in](#) [Next Challenge](#)

Test case 0

Compiler Message

Success

Test case 1

Test case 2

Test case 3

Test case 4

Test case 5

Input (stdin)

Download

Expected Output

Download

MEDIUM 05

```
from collections import Counter

n, m = map(int, input().split())
data = list(map(int, input().split()))
data_counter = Counter(data)
data_set = set(data)
set_a = set(map(int, input().split()))
set_b = set(map(int, input().split()))
happiness = 0
for i in data_set & set_a:
    happiness += data_counter[i]
for i in data_set & set_b:
    happiness -= data_counter[i]
print(happiness)
```

HackerRank | Prepare > Python > Sets > No Ideal

Problem

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.

Output Format

Output a single integer, your total happiness.

Sample Input

```
3 2
1 5 3
3 1
5 7
```

Sample Output

```
1
```

Line: 2 Col: 12

Upload Code as File Test against custom input

Run Code Submit Code

Congratulations

You solved this challenge. Would you like to challenge your friends? [f](#) [t](#) [in](#)

Next Challenge

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
```

MEDIUM 06

```
from collections import Counter, OrderedDict

class OrderedCounter(Counter, OrderedDict):
    pass

word_ar = []
n = int(input())
for i in range(n):
    word_ar.append(input().strip())
word_counter = OrderedCounter(word_ar)
print(len(word_counter))
for word in word_counter:
    print(word_counter[word], end=" ")
```

The screenshot displays the HackerRank interface for the 'Word Order' challenge. On the left, the problem description states: '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: $1 \leq n \leq 10^5$. 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. Sample Input: 4, bcdef, abcdefg, bcde, bcdef. Sample Output: 3, 2 1 1.

The main area shows a 'Congratulations' message: 'You solved this challenge. Would you like to challenge your friends?' with social media icons and a 'Next Challenge' button. Below this, a 'Success' panel displays the 'Input (stdin)' and 'Expected Output'.

Input (stdin):

```
4
bcdef
abcdefg
bcde
bcdef
```

Expected Output:

```
3
2 1 1
```

A list of test cases (0-6) is visible on the left, all marked as successful.

MEDIUM 07

```
import itertools

s = input().strip()
s_unique_element = list(set(s))
group = []
key = []
for k, g in itertools.groupby(s):
    group.append(list(g))
    key.append(k)
for i in range(len(group)):
    group_length = len(group[i])
    k = int(key[i])
    print((group_length, k), end=" ")
```

The screenshot displays the HackerRank interface for the 'Compress the String' challenge. The left sidebar contains navigation links: Problem, Submissions, Leaderboard, Discussions, and Tutorial. The main content area on the left provides the problem details:

- Problem:** 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](#).
- 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. $1 \leq |S| \leq 10^4$.
- Sample Input:** 12223111
- Sample Output:** (1, 1) (3, 2) (1, 3) (2, 1)
- Explanation:** First, the character 1 occurs only once. It is replaced by (1, 1). Then the character 2 occurs three times, and it is replaced by (3, 2) and so on. Also, note the single space within each compression and between the compressions.

The right sidebar shows the submission interface with buttons for 'Upload Code as File', 'Test against custom input', 'Run Code', and 'Submit Code'. A green banner displays 'Congratulations' and 'You solved this challenge. Would you like to challenge your friends?' with social media icons and a 'Next Challenge' button. Below this, a list of test cases (0-6) is shown, all marked as successful. The 'Compiler Message' section indicates 'Success'. The 'Input (stdin)' and 'Expected Output' sections show the sample input '12223111' and the expected output '(1, 1) (3, 2) (1, 3) (2, 1)' respectively.

MEDIUM 08

```
import re
n, m = input().strip().split(' ')
n, m = [int(n), int(m)]
matrix = []
for _ in range(n):
    matrix_t = str(input())
    matrix.append(matrix_t)
complete = ""
for el in zip(*matrix):
    complete += " ".join(el)
print(re.sub(r'(?<=\w) ([^\w]+) (?=\w)', " ", complete))
```

The screenshot shows the HackerRank interface for the 'Company Logo' problem. The left sidebar contains navigation links: Problem, Submissions, Leaderboard, Discussions, and Tutorial. The main content area is divided into several sections: Problem, Input Format, Constraints, Output Format, Sample Input 0, and Sample Output 0. The problem description states that a multinational brand is looking for the three most common characters in its company name. The input is a string S , and the output should be the three most common characters with their occurrence counts, sorted by count and then alphabetically. The sample input is 'aabbccde' and the sample output is 'b 3', 'a 2', 'c 2'. The right sidebar shows the test cases, all of which are passed. The compiler message indicates 'Success'.

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, 'aabbccde' would have its logo with the letters 'b', 'a', 'c'.

Input Format

A single line of input containing the string S .

Constraints

- $3 < \text{len}(S) \leq 10^4$
- S has at least 3 distinct characters

Output Format

Print the three most common characters along with their occurrence count each on a separate line. Sort output in descending order of occurrence count. If the occurrence count is the same, sort the characters in alphabetical order.

Sample Input 0

```
aabbccde
```

Sample Output 0

```
b 3
a 2
c 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 aabbccde
```

Expected Output

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

MEDIUM 09

```
from collections import deque

cas = int(input())
for _ in range(cas):
    n = int(input())
    dq = deque(map(int, input().split()))
    possible = True
    element = (2**31) + 1
    while dq:
        left_element = dq[0]
        right_element = dq[-1]
        if left_element >= right_element and element >= left_element:
            element = dq.popleft()
        elif right_element >= left_element and element >= right_element:
            element = dq.pop()
        else:
            possible = False
            break
    if possible:
        print("Yes")
    else:
        print("No")
```

The screenshot shows the HackerRank interface for the 'Piling Up!' challenge. On the left, the problem description states: '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.' It includes an example with blocks [1, 2, 3, 8, 7] resulting in 'No', and another with [1, 2, 3, 7, 8] resulting in 'Yes'. The input format specifies T test cases, each with n cubes and their side lengths. Constraints are $1 \leq T \leq 5$, $1 \leq n \leq 10^5$, and $1 \leq sideLength < 2^{31}$. The output format requires printing 'Yes' or 'No' for each test case.

The main area shows a 'Congratulations' message: 'You solved this challenge. Would you like to challenge your friends?' with social media icons and a 'Next Challenge' button. Below this, a list of test cases is shown, all marked as 'Success'. Test case 0 is expanded, showing the input (stdin) as '2', '6', '4 1 2 1 3 4', '1 3 2' and the expected output as 'Yes', 'No'.

MEDIUM 10

```
for i in range(1,int(input())+1): #More than 2 lines will result in
    0 score. Do not leave a blank line also
    print (((10**i - 1) // 9) ** 2)
```

The screenshot shows the HackerRank interface for the 'Triangle Quest 2' challenge. The left sidebar contains navigation links: Problem, Submissions, Leaderboard, Discussions, and Tutorial. The main content area on the left displays the problem description, which asks for a palindromic triangle of size N . It provides an example for $N=5$ and includes a note about scoring. The 'Input Format' section states that a single line of input containing the integer N is provided. The 'Constraints' section specifies $0 < N < 10$. The 'Output Format' section requires printing the palindromic triangle. A 'Sample Input' of 5 and its corresponding 'Sample Output' are shown. The right side of the interface shows the code editor with a Python solution, a 'Run Code' button, and a 'Submit Code' button. Below the code editor, a green 'Congratulations' banner indicates a successful submission. The 'Test case' results show all five test cases passed. The 'Compiler Message' section displays 'Success'. The 'Input (stdin)' section shows the input '5'. The 'Expected Output' section shows the expected palindromic triangle for $N=5$.

Problem

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:

```
1
121
12321
1234321
123454321
```

You can't take more than two lines. The first line (a for-statement) is already written for you.
You have to complete the code using exactly one print statement.

Note:
Using anything related to strings will give a score of 0.
Using more than one for-statement will give a score of 0.

Input Format
A single line of input containing the integer N .

Constraints
 $0 < N < 10$

Output Format
Print the palindromic triangle of size N as explained above.

Sample Input
5

Sample Output

Run Code **Submit Code**

Congratulations
You solved this challenge. Would you like to challenge your friends? [f](#) [t](#) [in](#) [Next Challenge](#)

Test case 0 **Test case 1** **Test case 2** **Test case 3** **Test case 4** **Test case 5**

Compiler Message
Success

Input (stdin)
5

Expected Output
1
121
12321
1234321
123454321

MEDIUM 11

```
from itertools import combinations

n = int(input())
ar = input().split()
k = int(input())
comb_list = list(combinations(ar, k))
a_list = [e for e in comb_list if "a" in e]
print(len(a_list) / len(comb_list))
```

The screenshot shows the HackerRank interface for a challenge titled 'Iterables and Iterators'. The left sidebar contains navigation links: Problem, Submissions, Leaderboard, Discussions, and Tutorial. The main content area on the left details the problem: it asks for the probability that at least one of K selected indices from a list of N lowercase letters contains the letter 'a'. It includes input/output formats, constraints ($1 \leq N \leq 10$, $1 \leq K \leq N$), and a sample input/output. The sample input is 4, 'a a c d', and 2, resulting in an expected output of 0.833333333333. The right side of the interface shows a code editor with the Python solution, a 'Run Code' button, and a 'Submit Code' button. Below the code editor, a green banner displays 'Congratulations' and a 'Next Challenge' button. A list of test cases (0-6) is shown, all marked as successful. The compiler message area at the bottom right confirms 'Success'.

MEDIUM 12

```
for i in range(1,int(input())): #More than 2 lines will result in 0
    score. Do not leave a blank line also
    print(i * ((10**i - 1) // 9))
```

The screenshot shows the HackerRank interface for the 'Triangle Quest' challenge. The left sidebar contains navigation links: Problem, Submissions, Leaderboard, Discussions, and Tutorial. The main content area on the left describes the problem: given a positive integer N , print a numerical triangle of height $N - 1$. It includes a sample input of 5 and a sample output showing a triangle of numbers 1 through 4444. The right panel shows the code editor with a Python solution, a 'Congratulations' message, and test case results. The test cases are all passed, and the compiler message is 'Success'.

Problem

You are given a positive integer N . Print a numerical triangle of height $N - 1$ like the one below:

```
1
22
333
4444
55555
.....
```

Can you do it using only **arithmetic operations, a single for loop and print statement**? Use no more than two lines. The first line (the for statement) is already written for you. You have to complete the print statement.

Note: Using anything related to strings will give a score of 0.

Input Format

A single line containing integer, N .

Constraints

$1 \leq N \leq 9$

Output Format

Print $N - 1$ lines as explained above.

Sample Input

```
5
```

Sample Output

```
1
22
333
4444
```

Run Code **Submit Code**

Congratulations

You solved this challenge. Would you like to challenge your friends? [f](#) [t](#) [in](#) [Next Challenge](#)

Test case 0 **Test case 1** **Test case 2**

Compiler Message

Success

Input (stdin) [Download](#)

```
5
```

Expected Output [Download](#)

```
1
22
333
4444
```

MEDIUM 13

```
import math
class Complex:
    def __init__(self, real, imaginary):
        self.real = real
        self.imaginary = imaginary

    def __add__(self, no):
        return Complex(self.real + no.real, self.imaginary + no.imaginary)

    def __sub__(self, no):
        return Complex(self.real - no.real, self.imaginary - no.imaginary)

    def __mul__(self, no):
        return Complex(
            self.real * no.real - self.imaginary * no.imaginary,
            self.real * no.imaginary + self.imaginary * no.real,
        )

    def __truediv__(self, no):
        divider = no.real**2 + no.imaginary**2
        return Complex(
            (self.real * no.real + self.imaginary * no.imaginary) /
divider,
            (self.imaginary * no.real - self.real * no.imaginary) /
divider,
        )

    def mod(self):
        return Complex(math.sqrt(self.real**2 + self.imaginary**2),
0.00)

    def __str__(self):
        if self.imaginary == 0:
            result = "%.2f+0.00i" % (self.real)
        elif self.real == 0:
            if self.imaginary >= 0:
                result = "0.00+%.2fi" % (self.imaginary)
            else:
                result = "0.00-%.2fi" % (-self.imaginary)
```

[Prepare](#) > [Python](#) > [Classes](#) > [Classes: Dealing with Complex Numbers](#)

Problem

Submissions

Leaderboard

Discussions

Tutorial

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.

The real and imaginary precision part should be correct up to two decimal places.

Input Format

One line of input: The real and imaginary part of a number separated by a space.

Output Format

For two complex numbers C and D , the output should be in the following sequence on separate lines:

- $C + D$
- $C - D$
- $C * D$
- C / D
- $mod(C)$
- $mod(D)$

For complex numbers with non-zero real(A) and complex part(B), the output should be in the following format:

$A + Bi$

Replace the plus symbol (+) with a minus symbol (-) when $B < 0$.

For complex numbers with a zero complex part i.e. real numbers, the output should be:

$A + 0.00i$

For complex numbers where the real part is zero and the complex part(B) is non-zero, the output should be:

$0.00 + Bi$

Sample Input

```
2 1
5 6
```

```

1         return result
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44         if __name__ == '__main__': ...

```

Line: 44 Col: 1

Upload Code as File

Test against custom Input

Run Code

Submit Code

Congratulations

You solved this challenge. Would you like to challenge your friends?

Next Challenge

Test case 2

Test case 3

Test case 4

Test case 5

Test case 6

Test case 7

Test case 8

Compiler Message

Success

Input (stdin)

Download

```

1 2 1
2 5 6

```

Expected Output

Download

```

1 7.00+7.00i
2 -3.00-5.00i
3 4.00+17.00i
4 0.26-0.11i

```

HackerRank
Prepare > Python > Classes > Classes: Dealing with Complex Numbers
Exit Full Screen View

Problem

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.

The real and imaginary precision part should be correct up to two decimal places.

Input Format

One line of input: The real and imaginary part of a number separated by a space.

Output Format

For two complex numbers C and D , the output should be in the following sequence on separate lines:

- $C + D$
- $C - D$
- $C * D$
- C / D
- $\text{mod}(C)$
- $\text{mod}(D)$

For complex numbers with non-zero real(A) and complex part(B), the output should be in the following format:
 $A + Bi$
 Replace the plus symbol (+) with a minus symbol (-) when $B < 0$.

For complex numbers with a zero complex part i.e. real numbers, the output should be:
 $A + 0.00i$

For complex numbers where the real part is zero and the complex part(B) is non-zero, the output should be:
 $0.00 + Bi$

Sample Input

```
2 1  
5 6
```

Submissions

Leaderboard

Discussions

Editorial

```
def complex_add(c1, c2):
    return result

44
> if __name__ == '__main__': ...
```

Line: 44 Col: 1

☐ Test against custom Input

Run Code

Submit Code

Congratulations

You solved this challenge. Would you like to challenge your friends? [f](#) [t](#) [in](#)

Next Challenge

Tests

✔

Test case 2

↕

✔

Test case 3

↕

✔

Test case 4

↕

✔

Test case 5

↕

✔

Test case 6

↕

✔

Test case 7

↕

✔

Test case 8

↕

Compiler Message

Success

Input (stdin)

Download

1	2 1
2	5 6

Expected Output

Download

1	7.00+7.00i
2	-3.00-5.00i
3	4.00+17.00i
4	0.26-0.11i

MEDIUM 14

```
import math
import os
import random
import re
import sys

if __name__ == "__main__":
    n, m = map(int, input().split())

    arr = []

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

    k = int(input())

    for i in sorted(arr, key=lambda x: x[k]):
        print(*i)
```

Problem

Submissions

Leaderboard

Discussions

Editorial

HackerRank

Prepare > Python > Built-Ins > Athlete Sort

Exit Full Screen View

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.

Input Format

The first line contains N and M separated by a space.

The next N lines each contain M elements.

The last line contains K .

Constraints

$1 \leq N, M \leq 1000$
 $0 \leq K < M$
Each element ≤ 1000

Output Format

Print the N lines of the sorted table. Each line should contain the space separated elements. Check the sample below for clarity.

Sample Input 0

Line: 20 Col: 1

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

Congratulations

You solved this challenge. Would you like to challenge your friends? [f](#) [t](#) [in](#) [Next Challenge](#)

Test case 0

Test case 1

7 1 0

9 9 9

1 23 12

6 5 9

1

Expected Output

Download

7 1 0

10 2 5

6 5 9

9 9 9

1 23 12

MEDIUM 15

```
s = input()
print(
    "".join(
        sorted(
            s,
            key=lambda x: (
                x.isdigit() and int(x) % 2 == 0,
                x.isdigit(),
                x.isupper(),
                x.islower(),
                x,
            ),
        )
    )
)
```

The screenshot shows the HackerRank interface for the 'ginortS' challenge. The left sidebar contains navigation links: Problem, Submissions, Leaderboard, Discussions, and Tutorial. The main content area on the left describes the problem: given a string *S*, sort it such that lowercase letters are first, followed by uppercase letters, then even digits, and finally odd digits. It includes sample input 'Sorting1234' and sample output 'ginortS1324'. The right panel shows the code editor with a Python solution, a 'Run Code' button, and a 'Submit Code' button. Below the editor, a green 'Congratulations' banner indicates the challenge was solved. The bottom section displays test cases 0 through 5, all of which passed, with the input 'Sorting1234' and the expected output 'ginortS1324'.

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*.

Sample Input

```
Sorting1234
```

Sample Output

```
ginortS1324
```

Compiler Message

Success

Test case 0

Test case 1

Test case 2

Test case 3

Test case 4

Test case 5

Input (stdin)

```
Sorting1234
```

Expected Output

```
ginortS1324
```

Next Challenge

MEDIUM 16

```
import re

def fun(s):
    return re.search(r"^[w-]+@[a-zA-Z0-9]+\.[a-zA-Z]{1,3}$", s)

def filter_mail(emails):
    return list(filter(fun, emails))

if __name__ == '__main__':
    n = int(input())
    emails = []
    for _ in range(n):
        emails.append(input())

filtered_emails = filter_mail(emails)
filtered_emails.sort()
print(filtered_emails)
```

The screenshot shows the HackerRank interface for the challenge 'Validating Email Addresses With a Filter'. The left sidebar contains navigation links: Problem, Submissions, Leaderboard, Discussions, and Tutorial. The main content area on the left provides the problem description, rules for valid email addresses, a concept of the filter function, and an example. The right side of the interface shows the code editor with the solution code, a 'Congratulations' message, a 'Next Challenge' button, and a list of test cases (Test case 3 through Test case 9) all marked as successful. Below the test cases, the 'Compiler Message' shows 'Success', and the 'Input (stdin)' and 'Expected Output' are displayed.

Problem

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.

Submissions

Line: 7 Col: 1

Upload Code as File Test against custom input

Run Code Submit Code

Congratulations

You solved this challenge. Would you like to challenge your friends? [f](#) [t](#) [in](#) [Next Challenge](#)

Compiler Message

Success

Input (stdin) [Download](#)

```
1 3
2 tara@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']
```

Test cases

- Test case 3 [🔗](#)
- Test case 4 [🔗](#)
- Test case 5 [🔗](#)
- Test case 6 [🔗](#)
- Test case 7 [🔗](#)
- Test case 8 [🔗](#)
- Test case 9 [🔗](#)

MEDIUM 17

```
from fractions import Fraction
from functools import reduce
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)
```

The screenshot shows the HackerRank interface for the 'Reduce Function' challenge. The left sidebar contains navigation links: Problem, Submissions, Leaderboard, Discussions, and Tutorial. The main content area displays the problem description, which asks to find the product of a list of rational numbers. It includes a 'Concept' section explaining the `reduce()` function, an 'Input Format' section, and an 'Output Format' section. The right sidebar shows the submission results, including a 'Congratulations' message, a 'Next Challenge' button, and a list of test cases. The bottom right panel shows the 'Compiler Message' and 'Input (stdin)' for the test cases.

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
```

Input Format

First line contains n , the number of rational numbers.

The i^{th} of next n lines contain two integers each, the numerator (N_i) and denominator (D_i) of the i^{th} rational number in the list.

Constraints

- $1 \leq n \leq 100$
- $1 \leq N_i, D_i \leq 10^9$

Output Format

Print only one line containing the numerator and denominator of the product of the numbers in

Submissions

Upload Code as File Test against custom Input

Run Code Submit Code

Success

You have earned 30.00 points!
29/115 challenges solved. 25%

Congratulations

You solved this challenge. Would you like to challenge your friends? [f](#) [t](#) [in](#)

Next Challenge

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 21
2 684025282 932952183
3 349232934 278093065
4 778706161 742081687
5 374870211 874099626
6 849763633 211127281
7 566205501 508794028
8 814324820 443967409
9 402053385 120666811
```

MEDIUM 18

```
import re
import sys

n = int(input())
for line in sys.stdin:
    remove_and = re.sub(r"(?<= )(&&)(?= )", "and", line)
    remove_or = re.sub(r"(?<= )(\||\|)(?= )", "or", remove_and)
    print(remove_or, end="")
```

The screenshot displays the HackerRank interface for a challenge titled "Regex Substitution". The left sidebar contains navigation links: Problem, Submissions, Leaderboard, Discussions, and Portal. The main content area is divided into two sections. The top section, "Transformation of Strings", provides a description of the `re.sub()` method and includes a code editor with the following Python 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")
```

The output of this code is shown as "1 4 9 16 25 36 49 64 81". The bottom section, "Replacements in Strings", shows a code editor with HTML code that uses `re.sub()` to replace `<script>` with `</script>`.

```
import re

html = """
<head>
<title>HTML</title>
</head>
<object type="application/x-flash"

```

The right sidebar shows the challenge's progress: "You have earned 20.00 points! 31/115 challenges solved. 27%". A green banner reads "Congratulations You solved this challenge. Would you like to challenge your friends?" with social media icons and a "Next Challenge" button. Below this, a list of test cases (0-6) is shown, all marked as successful. The "Compiler Message" section displays a "Success" message and the input (stdin) for the challenge:

```
1 11
2 a = 1;
3 b = input();
4
5 if a + b > 0 && a - b < 0:
6     start()
7 elif a+b > 10 || a/b < 1:
8     stop()
9 print set(list(a)) | set(list(b))
```

MEDIUM 19

```
import re

n = int(input())
for _ in range(n):
    credit = input().strip()
    credit_removed_hiphen = credit.replace("-", "")
    valid = True
    length_16 = bool(re.match(r"^[4-6]\d{15}$", credit))
    length_19 = bool(re.match(r"^[4-6]\d{3}-\d{4}-\d{4}-\d{4}$", credit))
    consecutive = bool(re.findall(r"(?=(\d)\1\1\1)", credit_removed_hiphen))
    _hiphen)
    if length_16 == True or length_19 == True:
        if consecutive == True:
            valid = False
        else:
            valid = False
    if valid:
        print("Valid")
    else:
        print("Invalid")
```

The screenshot shows the HackerRank interface for the 'Validating Credit Card Numbers' challenge. The left sidebar contains navigation links: Problem, Submissions, Leaderboard, Discussions, and Profile. The main content area on the left provides the problem description, characteristics of a valid credit card, examples of valid and invalid numbers, and the input format. The right panel shows the submission interface with buttons for 'Run Code' and 'Submit Code', a progress bar indicating 26% completion (30/115 challenges solved), a 'Congratulations' message, and a list of test cases with their expected outputs.

Problem Description: 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:

- ▶ 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.

Examples:

Valid Credit Card Numbers

```
4253625879615786
442444424442444
5122-2368-7954-3214
```

Invalid Credit Card Numbers

```
42536258796157867 #17 digits in card number → Invalid
4424444424442444 #Consecutive digits are repeating 4 or more times → Invalid
5122-2368-7954 - 3214 #Separators other than '-' are used → Invalid
44244x4424442444 #Contains non digit characters → Invalid
0525362587961578 #Doesn't start with 4, 5 or 6 → Invalid
```

Input Format

The first line of input contains an integer N .

The next N lines contain credit card numbers.

Submission Interface:

- Buttons: Upload Code as File, Test against custom input, Run Code, Submit Code.
- Progress: You have earned 40.00 points! 30/115 challenges solved. 26%
- Message: Congratulations. You solved this challenge. Would you like to challenge your friends?
- Test Cases:

Test Case	Input	Expected Output
Test case 0	4253625879615786	Valid
Test case 1	61234-567-8912-3456	Valid
Test case 2	4123356789123456	Invalid
Test case 3	5133-3367-8912-3456	Valid
Test case 4	5123 - 3567 - 8912 - 3456	Invalid
Test case 5		Invalid

MEDIUM 20

```
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 = sum(1 for letter in word if is_vowel(letter))  
        score += 2 if num_vowels % 2 == 0 else 1  
    return score  
  
n = int(input())  
words = input().split()  
print(score_words(words))
```

The screenshot displays the HackerRank interface for the 'Words Score' challenge. On the left, the 'Problem' tab is active, showing the task: to debug the `score_words` function. The problem description states that vowels are 'a', 'e', 'i', 'o', 'u', and 'y'. The function takes a list of lowercase words and returns a score. The score of a single word is 2 if it contains an even number of vowels, and 1 otherwise. The total score is the sum of individual word scores. Constraints include $1 \leq n \leq 20$ and words being at most 20 letters long. The 'Sample Input 0' is '2 hacker book' and the 'Sample Output 0' is '4'. On the right, the submission result is shown. It indicates that the user has earned 10.00 points and solved 32/115 challenges. A 'Congratulations' message is displayed, and a list of test cases (0-6) is shown, all of which passed. The 'Compiler Message' section shows a 'Success' status. The 'Input (stdin)' section shows the input '2' and 'hacker book', and the 'Expected Output' section shows the output '4'.

MEDIUM 21

```
class EvenStream(object):
    def __init__(self):
        self.current = 0

    def get_next(self):
        to_return = self.current
        self.current += 2
        return to_return

class OddStream(object):
    def __init__(self):
        self.current = 1

    def get_next(self):
        to_return = self.current
        self.current += 2
        return to_return
```

The screenshot displays the HackerRank interface for a challenge. On the left, the 'Problem' tab is active, showing the task: to debug existing code to successfully execute all provided test files. It explains the concept of default argument values in Python and provides a code stub for a function `print_from_stream` that takes `n` and a `stream` object. The `stream` object has a `get_next()` method that returns the next value in a sequence of even or odd numbers. The right side of the interface shows the submission results. A green banner indicates 'Congratulations' and 'You have earned 30.00 points!'. Below this, a list of test cases (0 to 6) all show a 'Success' status. The 'Input (stdin)' and 'Expected Output' sections show the sequence of numbers generated by the stream.

Problem Description:

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`

Submission Results:

You have earned 30.00 points!
34/115 challenges solved. 30%

Congratulations
You solved this challenge. Would you like to challenge your friends?

Test Cases:

- Test case 0: Success
- Test case 1: Success
- Test case 2: Success
- Test case 3: Success
- Test case 4: Success
- Test case 5: Success
- Test case 6: Success

Input (stdin):

```
3
odd 2
even 3
odd 5
```

Expected Output:

```
1
3
```

MEDIUM 22

```
def minion_game(string):  
  
    s_length = len(s)  
    vowel_list = ["A", "E", "I", "O", "U"]  
    stuart_point = 0  
    kevin_point = 0  
    for i in range(s_length):  
        if s[i] in vowel_list:  
            kevin_point += s_length - i  
        else:  
            stuart_point += s_length - i  
    if stuart_point == kevin_point:  
        print("Draw")  
    elif kevin_point > stuart_point:  
        print("Kevin", kevin_point)  
    else:  
        print("Stuart", stuart_point)  
  
if __name__ == '__main__':  
    s = input()  
    minion_game(s)
```

The screenshot displays the HackerRank interface for the 'The Minion Game' challenge. On the left, the 'Problem' tab is active, showing the game rules and an example with the string 'BANANA'. The rules state that Stuart can only use consonants and Kevin can only use vowels to form substrings. The example shows that Kevin's vowel beginning word 'ANA' occurs twice in 'BANANA', earning him 2 points. Below the text is a diagram of the word 'BANANA' with subwords for Stuart (B, N, BA, NA, BAN) and Kevin (A, AN, ANA, ANAN, ANANA) listed with their respective scores. The main area shows a 'Congratulations' message for solving the challenge. On the right, the 'Test Cases' panel shows 6 test cases, all passed. The 'Compiler Message' panel shows a 'Success' status. The 'Input (stdin)' panel shows the input 'BANANA' and the 'Expected Output' panel shows 'Stuart 12'.

Problem

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.

For better understanding, see the image below:

STUART		KEVIN	
WORDS	SCORE	WORDS	SCORE
B	1	A	1
N	2	AN	2
BA	1	ANA	3
NA	2	ANAN	4
BAN	1	ANANA	5

Congratulations
You solved this challenge. Would you like to challenge your friends?

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)
BANANA

Expected Output
Stuart 12

HARD 01

```
import re
```

```
p = input().strip()
range_check = bool(re.match(r"^[1-9][0-9]{5}$", p))
repeat_check = len(re.findall(r"(?=[0-9])[0-9]\1", p))
print(range_check == True and repeat_check < 2)
```

```
import re
```

```
P = input()
```

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

The screenshot shows the HackerRank interface for a challenge titled "Validating Postal Codes" under the "Regex and Parsing" category. The problem description states that a valid postal code P must be a number from 100000 to 999999 inclusive and must not contain more than one alternating repetitive digit pair. Examples of alternating repetitive digits are provided: 121426 (1 is alternating), 523563 (no alternating), and 552523 (both 2 and 5 are alternating). The task is to provide two regular expressions: `regex_integer_in_range` and `regex_alternating_repetitive_digit_pair`. The provided code template uses `re.match` and `re.findall` to validate the input string `P`.

The right side of the screenshot shows the submission results. The user has earned 80.00 points for solving 36 out of 115 challenges (31%). A green banner displays "Congratulations" and a "Next Challenge" button. Below, the test cases are listed, all of which passed. The compiler message for the first test case is "Success". The input for the first test case is "110000" and the expected output is "False".

Test Case	Input (stdin)	Expected Output	Status
Test case 0	110000	False	Passed
Test case 1			Passed
Test case 2			Passed
Test case 3			Passed
Test case 4			Passed
Test case 5			Passed
Test case 6			Passed

HARD 02

```
import re

n, m = map(int, input().split())
character_ar = [""] * (n * m)
for i in range(n):
    line = input()
    for j in range(m):
        character_ar[i + (j * n)] = line[j]
decoded_str = "".join(character_ar)
final_decoded_str = re.sub(
    r"(?<=[A-Za-z0-9]) ([ !@#$%&]+) (?<=[A-Za-z0-9])", " ", decoded_str
)
print(final_decoded_str)
```

The screenshot shows the HackerRank interface for the 'Matrix Script' challenge. On the left, the problem description is visible, including a sample input matrix and its decoded output. The main area displays a success message: 'You have earned 100.00 points!' and '37/115 challenges solved.' Below this is a 'Congratulations' banner. On the right, there is a list of test cases (Test case 0 to Test case 6) and a 'Compiler Message' section showing 'Success'. The input for the test case is displayed as follows:

```
7 3
Tsi
hix
i#
sM
$a
#t%
ir!
```

HARD 03

```
import itertools

k, m = map(int, input().split())

main_ar = []
for _ in range(k):
    ar = list(map(int, input().split()))
    main_ar.append(ar[1:])

all_combination = itertools.product(*main_ar)
result = 0
for single_combination in all_combination:
    result = max(sum(x * x for x in single_combination) % m, result)
print(result)
```

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.

Constraints

- $1 \leq K \leq 7$
- $1 \leq M \leq 1000$
- $1 \leq N_i \leq 7$
- $1 \leq \text{Magnitude of elements in list} \leq 10^9$

Output Format

Output a single integer denoting the value S_{max} .

Sample Input

```
3 1000
2 5 4
3 7 8 9
```

Test Cases

- Test case 0
- Test case 1
- Test case 2
- Test case 3
- Test case 4
- Test case 5
- Test case 6

Input (stdin)

```
7 867
7 6429964 4173738 9941618 2744666 5392018 5813128 9452095
7 6517823 4135421 6418713 9924958 9370532 7948650 2027017
7 1506500 3460933 1550284 3679489 4538773 5216621 5645660
7 7443563 5181142 8804416 8726696 5358847 7155276 4433125
7 2230555 3920370 7851992 1176871 610460 309961 3921536
7 8518829 8639441 3373630 5936651 5291213 2308694 7477960
7 7178097 249343 9504976 8684596 6226627 1055259 4880436
```

Expected Output

```
866
```

Congratulations

You have earned 50.00 points!
38/115 challenges solved.

You solved this challenge. Would you like to challenge your friends?

Next Challenge