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:

HackerRank | Prepare > Python > Introduction > Write a function

Exit Full Screen View

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.

Submissions

Leaderboard

Change Theme Language Python 3

```python
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) Download

```
2000
```

Expected Output Download

```
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

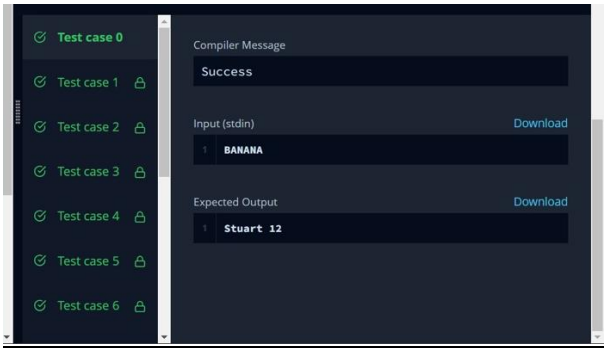
- string: the winner's name and score, separated by a space on one line, or `Draw` if there is no winner

Code:

```python
def minion_game(string):
    # your code goes here
    vowel = 'aeiou'.upper()
    strl = len(string)
    kevin = sum(strl-i for i in range(strl) if string[i] in vowel)
    stuart = strl*(strl + 1)/2 - kevin

    if kevin == stuart:
        print('Draw')

    elif kevin > stuart:
        print('Kevin %d' % kevin)

    else:
        print('Stuart %d' % stuart)

if __name__ == '__main__':
    s = input()
    minion_game(s)
```

Line: 16 Col: 1

Test Cases:



3. Merge the Tools!

Task:

Problem

Consider the following:

- A string, s, of length n where $s = c_0c_1 \ldots c_{n-1}$.
- An integer, k, where k is a factor of n.

We can split s into $\frac{n}{k}$ substrings where each subtring, 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

$s = \text{'AAABCADDE'}$

$k = 3$

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

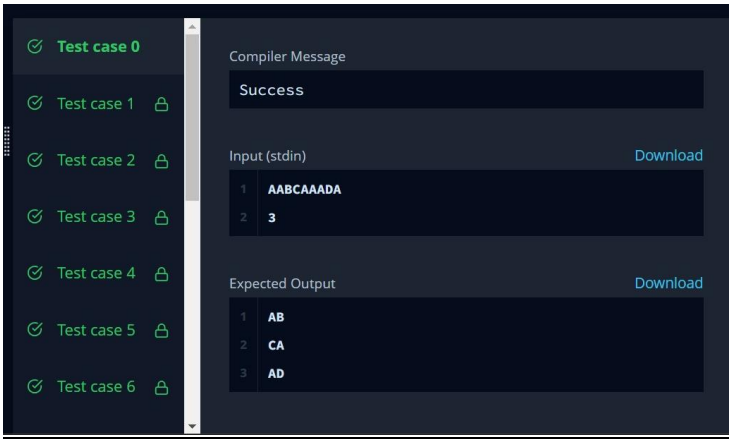
The first line contains a single string, s.

The second line contains an integer, k, the length of each substring.

Code:

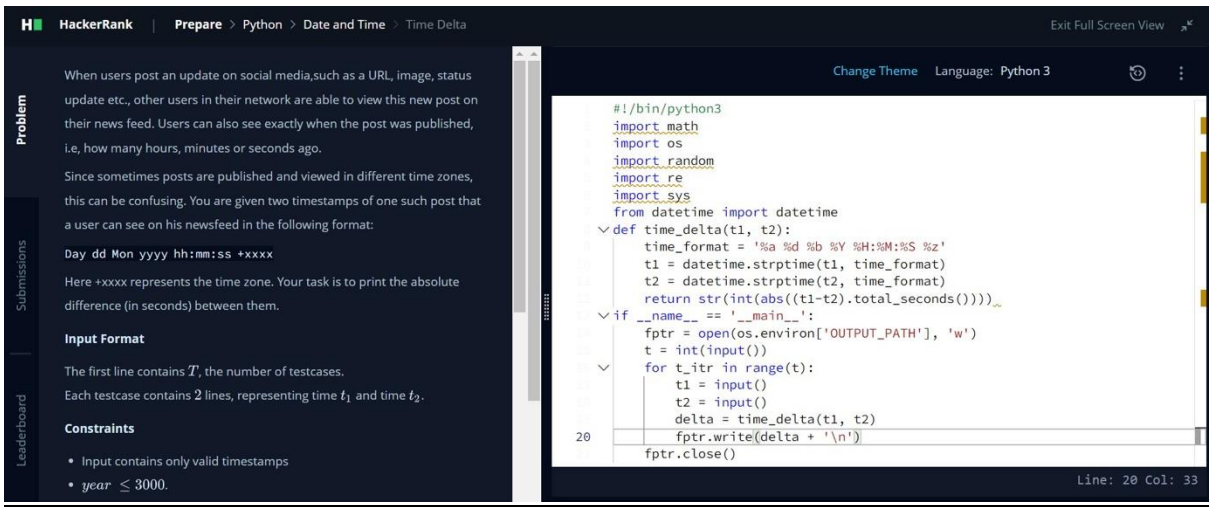
```python
def merge_the_tools(string, k):
    # your code goes here

    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:

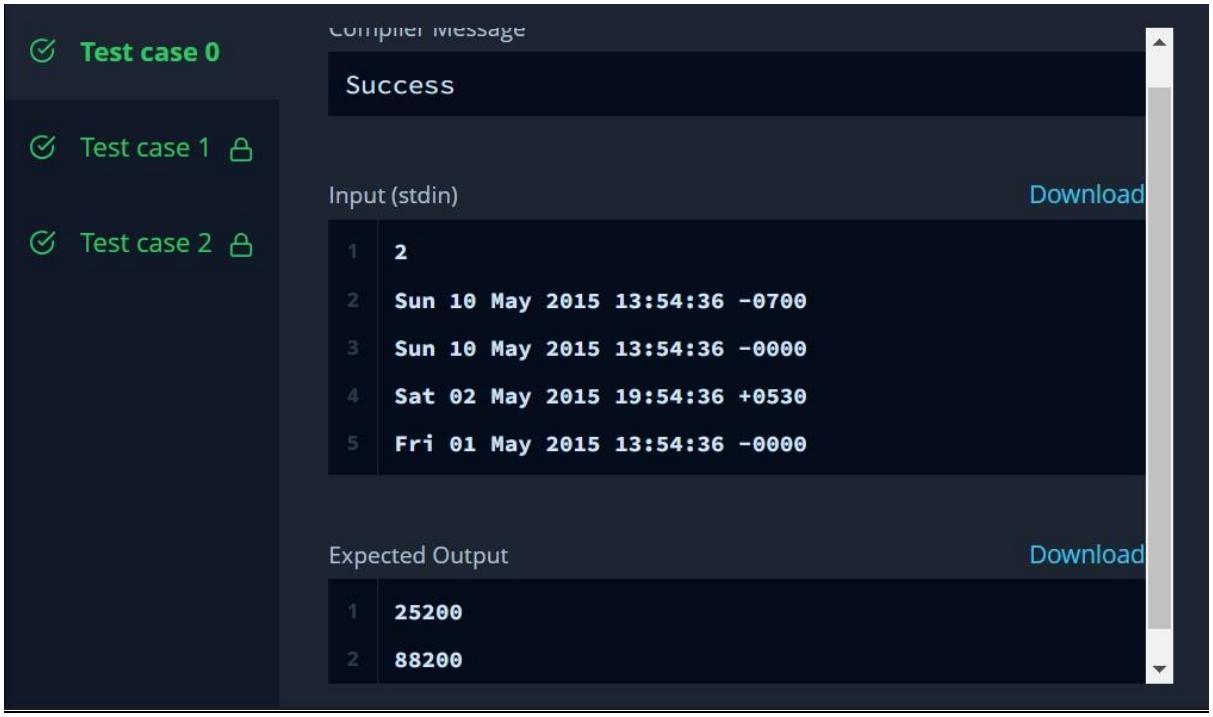


4. Time Delta

Task and Code:

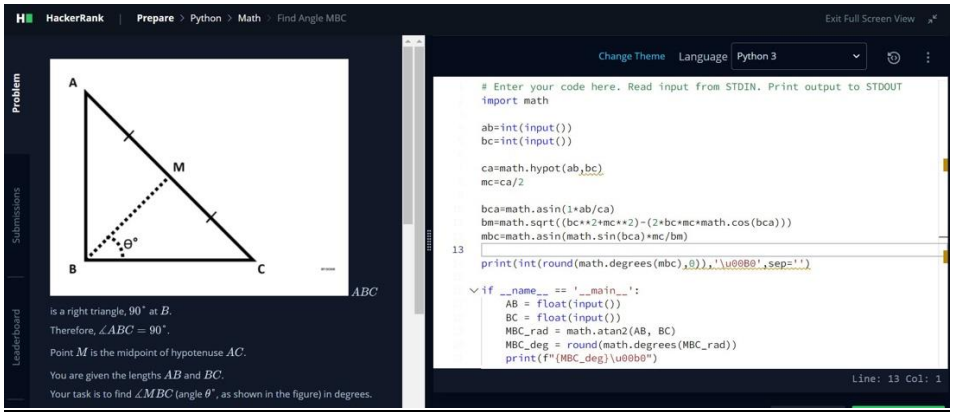


Test Cases:

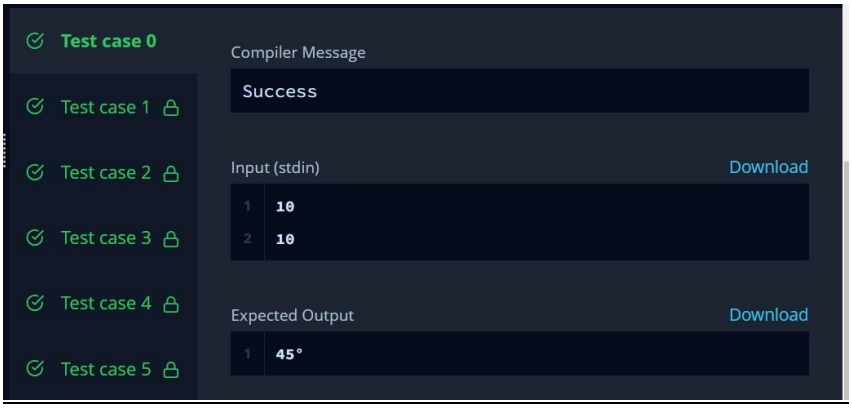


5. Find Angle MBC

Task and Code:



Test Cases:



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

$1 \leq n \leq 10^5$

$1 \leq m \leq 10^5$

$1 \leq Any\ integer\ in\ the\ input \leq 10^9$

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.

Change Theme Language Python 3

```python
# 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) Download

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

Expected Output Download

```
1
```

7. Word Order

Task and Code:

HackerRank | Prepare > Python > Collections > Word Order

Exit Full Screen View

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:

$1 \leq n \leq 10^5$

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.

Submissions

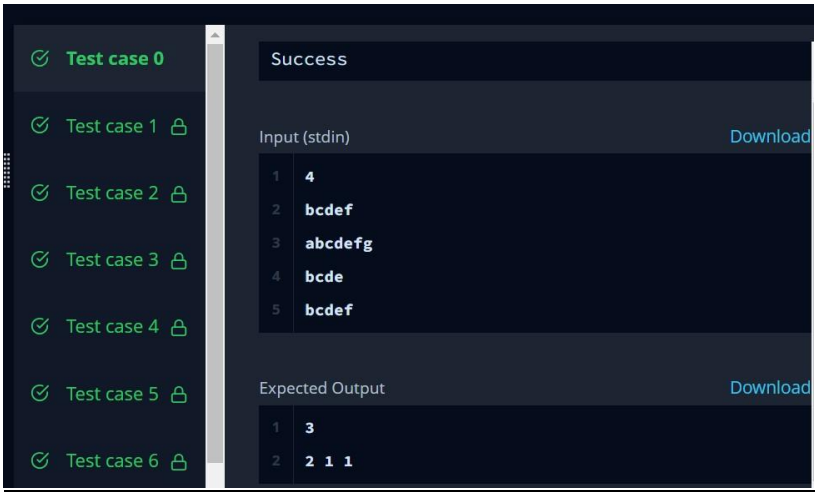
Leaderboard

Change Theme Language Python 3

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

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

words_occurences={}

for word in words:
    words_occurences[word]=0

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

print(len(words_occurences))

occurences=words_occurences.values()

for i in occurences:
    print(i, end=" ")
```

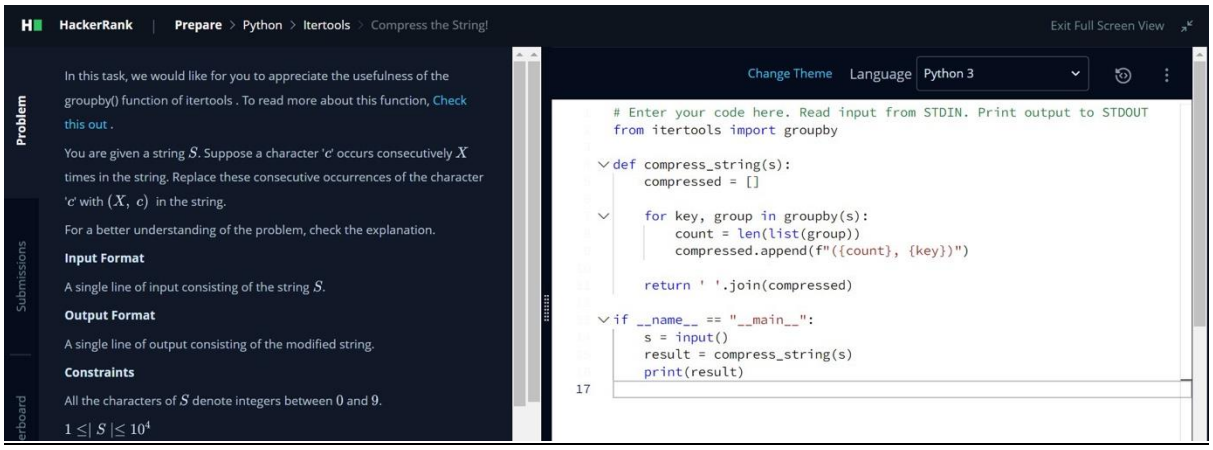
Line: 20 Col: 1

Test Cases:

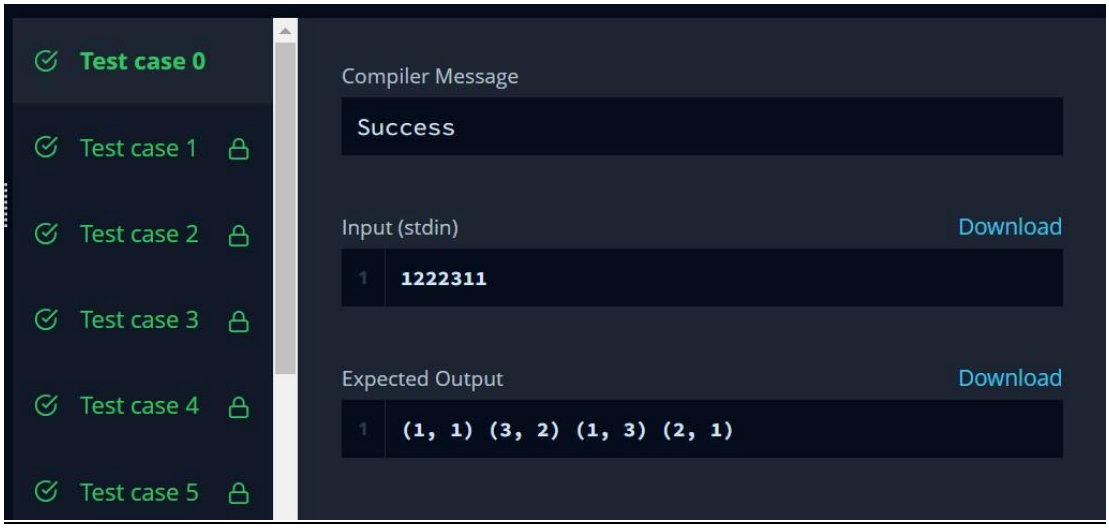


8. Compress the String!

Task and Code:



Test Cases:



9. Company Logo

Task and Code:

HackerRank | Prepare > Python > Collections > Company Logo

Exit Full Screen View

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, E.

Input Format

A single line of input containing the string S.

Submissions

Leaderboard

Change Theme Language Python 3

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

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

```
aabbbccde
```

Expected Output Download

```
b 3
a 2
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 $sideLength[j] \geq sideLength[i]$.

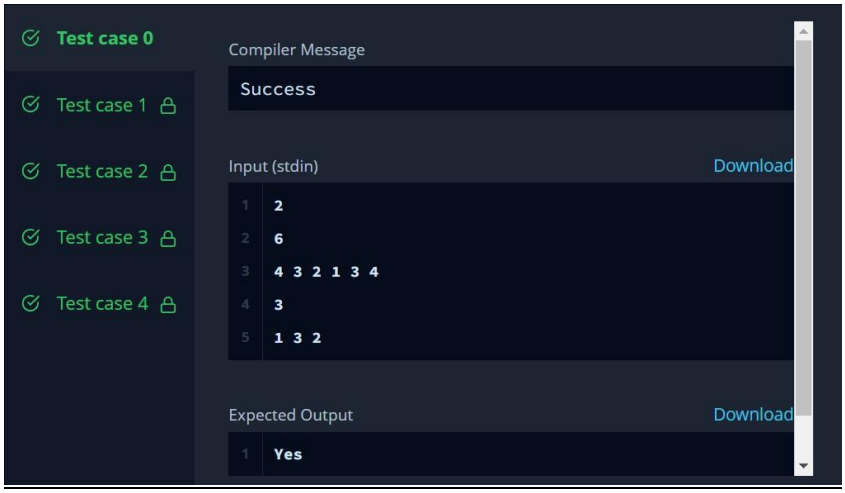
When stacking the cubes, you can only pick up either the leftmost or the rightmost cube each time. Print `Yes` if it is possible to stack the cubes. Otherwise, print `No`.

Code:

```python
def can_stack_cubes(n, cubes):
    left_index = 0
    right_index = n - 1
    top_cube = float('inf')

    while left_index <= right_index:
        if cubes[left_index] >= cubes[right_index] and cubes[left_index] <= top_cube:
            top_cube = cubes[left_index]
            left_index += 1

        elif cubes[right_index] >= cubes[left_index] and cubes[right_index] <= top_cube:
            top_cube = cubes[right_index]
            right_index -= 1

        else:
            return "No"

    return "Yes"
```

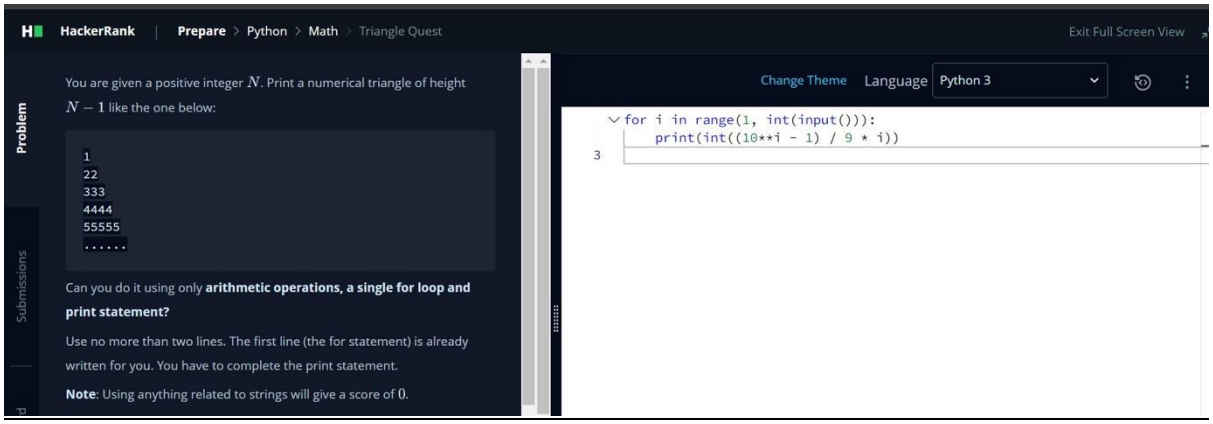
```python
    return "Yes"

if __name__ == "__main__":

    T = int(input())

    for _ in range(T):
        n = int(input())
        cubes = list(map(int, input().split()))
        result = can_stack_cubes(n, cubes)
        print(result)
```

Test Cases:

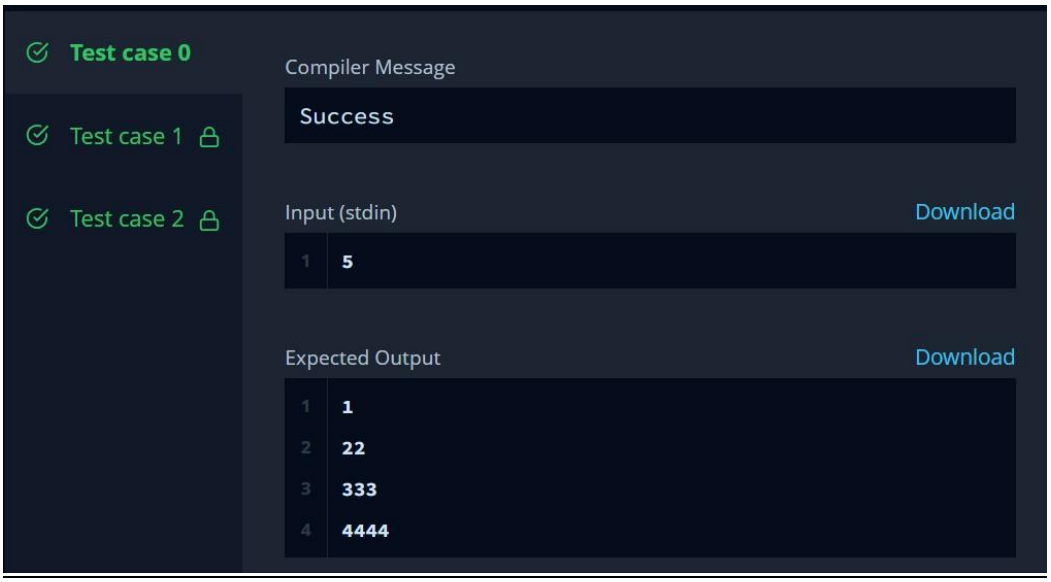


11. Triangle Quest

Task and Code:

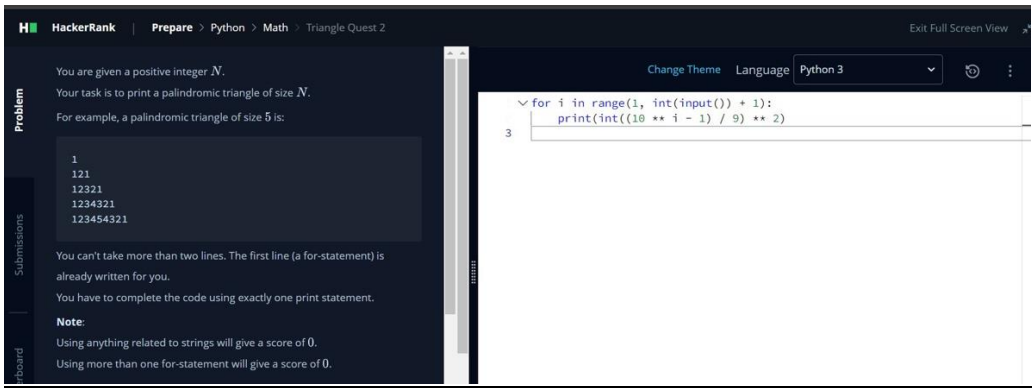


Test Cases:

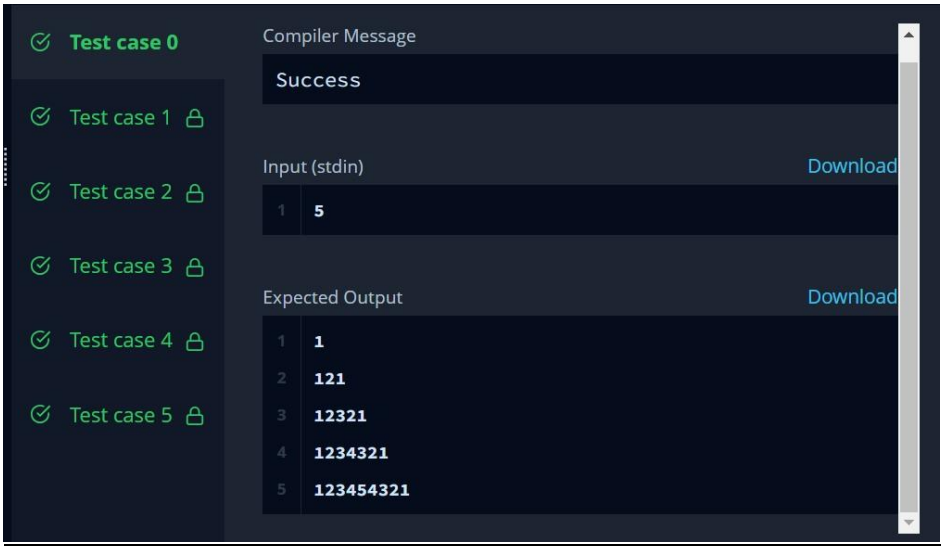


12. Triangle Quest 2

Task and Code:

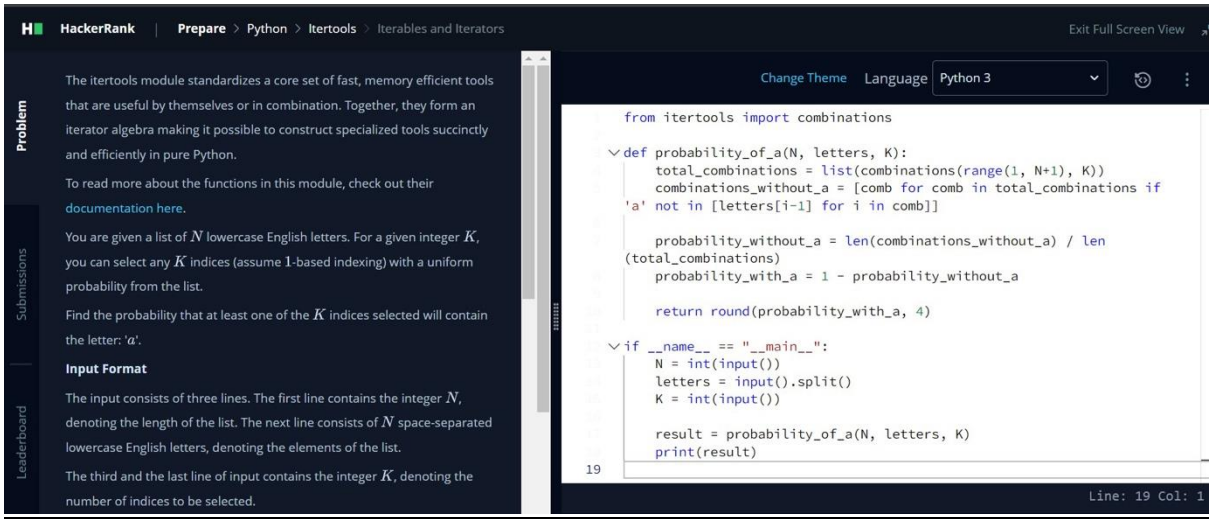


Test Cases:

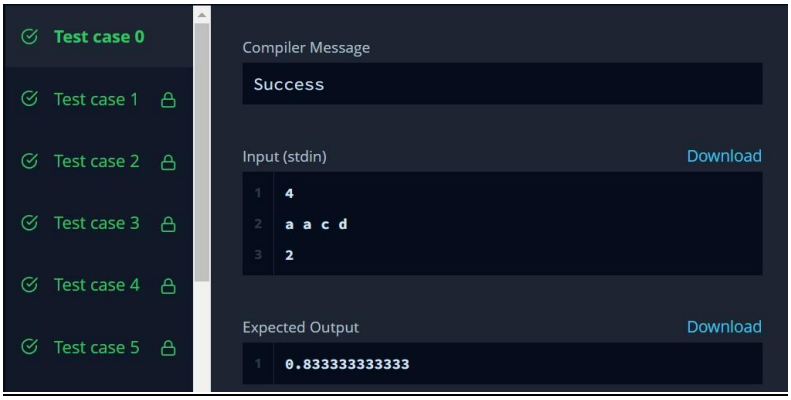


13. Iterables and Iterators

Task and Code:



Test Cases:



14. Classes: Dealing With Complex Numbers

Task:

Problem

Submissions

Leaderboard

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
- $mod(C)$
- $mod(D)$

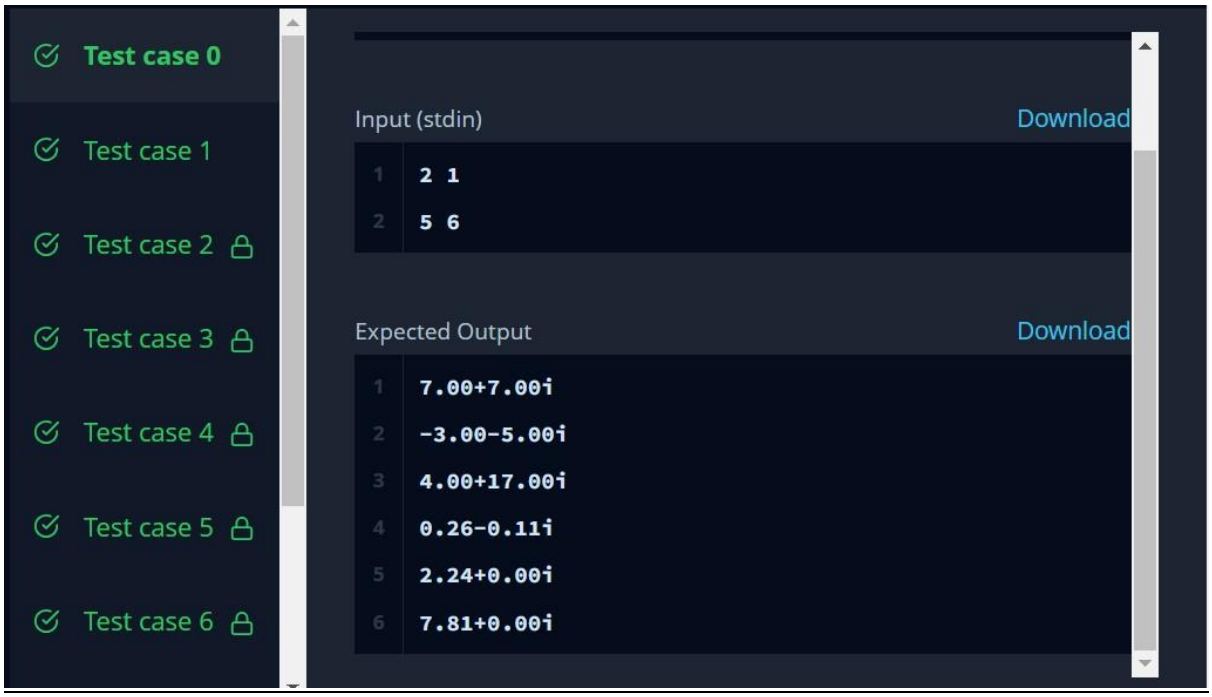
Code:

```
import math

class Complex(object):
    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):
        denominator = no.real**2 + no.imaginary**2
        real_part = (self.real * no.real + self.imaginary * no.imaginary)
/ denominator
```

```python
        imag_part = (self.imaginary * no.real - self.real * no.imaginary) / denominator
        return Complex(real_part, imag_part)

    def mod(self):
        return Complex(math.sqrt(self.real**2 + self.imaginary**2), 0)

    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" % (abs(self.imaginary))
        elif self.imaginary > 0:
            result = "%.2f+%.2fi" % (self.real, self.imaginary)
        else:
            result = "%.2f-%.2fi" % (self.real, abs(self.imaginary))
        return result
```

```python
if __name__ == '__main__':
    c = map(float, input().split())
    d = map(float, input().split())
    x = Complex(*c)
    y = Complex(*d)
    print(*map(str, [x+y, x-y, x*y, x/y, x.mod(), y.mod()]), sep='\n')
```

Test Cases:



15. Athlete Sort

Task and Code:

HackerRank | Prepare > Python > Built-Ins > Athlete Sort

Exit Full Screen View

Problem

Submissions

Leaderboard

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 Kth 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	sort based on k=1	4	26	195
3	41	188	→	1	32	190
4	26	195	i.e (age)	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 atheletes are of the same age, print the row that appeared first in the input.

Change Theme Language Python 3

```python
#!/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

```
5 3
10 2 5
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
```

16. ginortS

Task and Code:

HackerRank | Prepare > Python > Built-Ins > ginortS

Exit Full Screen View

Problem

Submissions

Leaderboard

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

- $0 < len(S) < 1000$

Output Format

Output the sorted string S.

Change Theme Language Python 3

```python
# 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

Test case 1

Test case 2

Test case 3

Test case 4

Compiler Message

Success

Input (stdin) Download

```
Sorting1234
```

Expected Output Download

```
ginortS1324
```

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

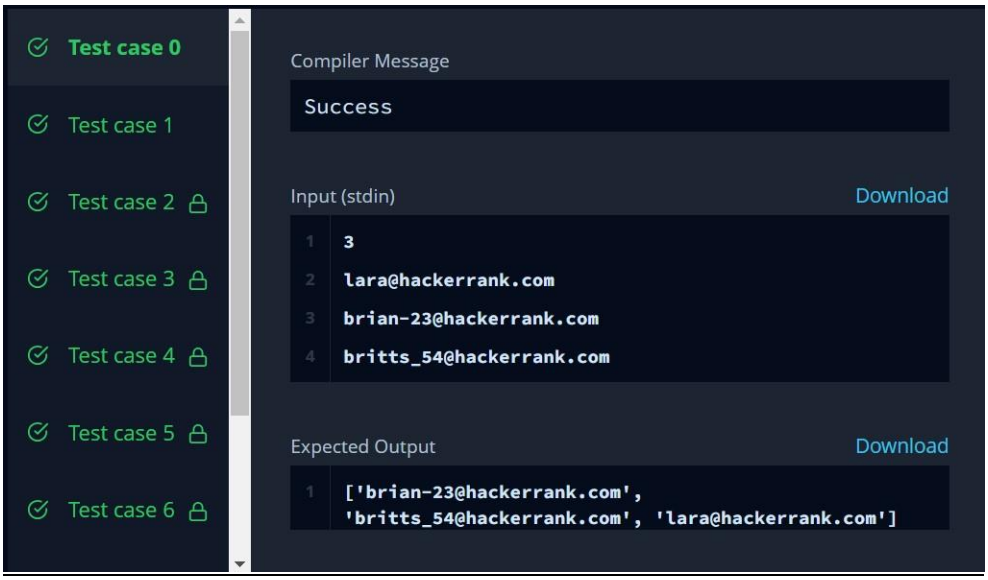
Returns

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

Code:

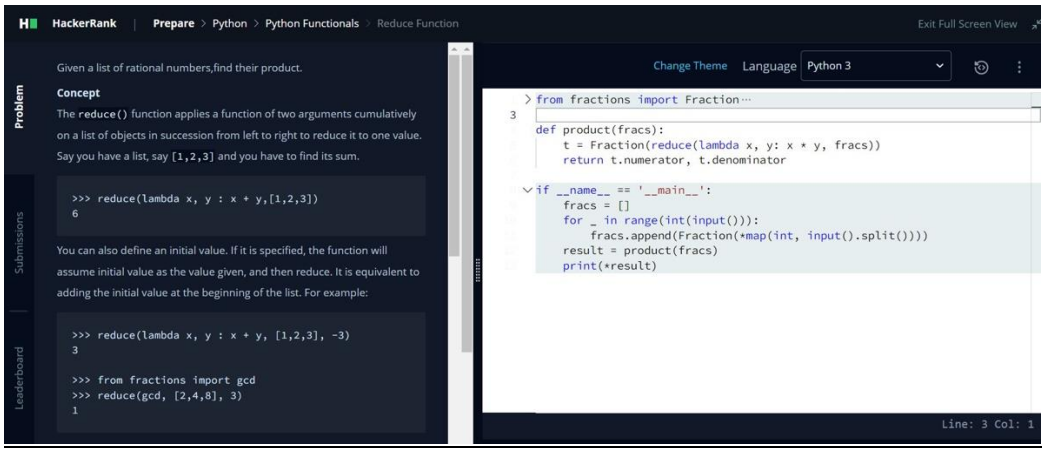
```python
def fun(s):

    if '@' not in s:
        return False

    parts = s.split('@')

    if len(parts) != 2:
        return False

    username, rest = parts

    if not username.replace('_', '').replace('-', '').isalnum():
        return False

    if '.' not in rest:
        return False

    website, extension = rest.split('.')

    if not (website.isalnum() and extension.isalpha() and len
(extension) <= 3):
        return False

    return True

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)
```

Test Cases:

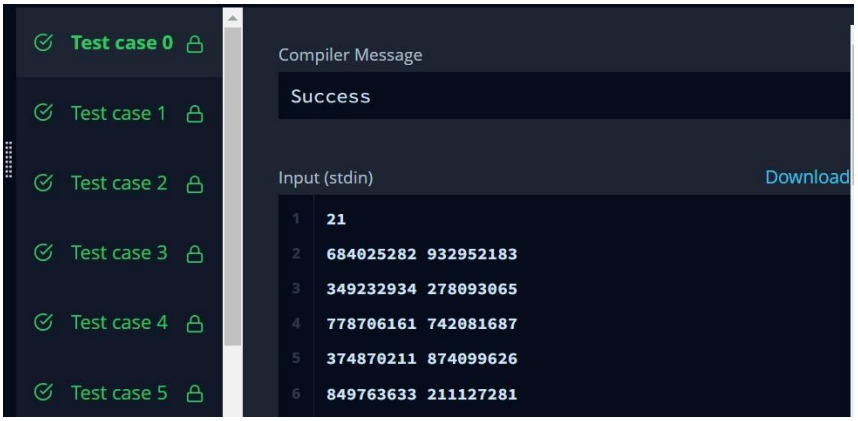


18. Reduce Function

Task and Code:



Test Cases:



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 Run Code Submit Code

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 Download

```
a = 1;
b = input();

if a + b > 0 and a - b < 0:
    start()
elif a*b > 10 or a/b < 1:
    stop()
print set(list(a)) | set(list(b))
#Note do not change &&& or ||| or & or |
#Only change those '&&' which have space on both sides.
#Only change those '|| which have space on both sides.
```

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:

- ▶ 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.

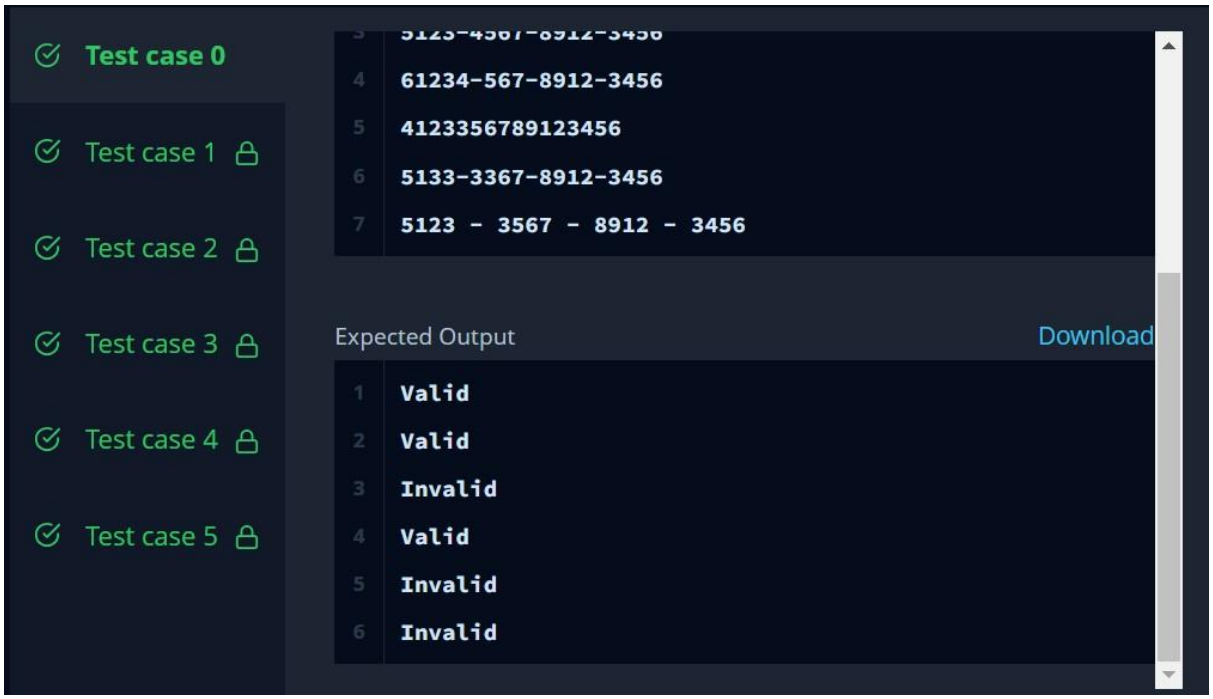
Submissions

Code:

```python
def is_valid(card_number):
    if not (16 == len(card_number) or (19 == len(card_number) and card_number[4] == '-' and card_number[9] == '-' and card_number[14] == '-')):
        return "Invalid"

    if card_number[0] not in '456':
        return "Invalid"

    card_number = card_number.replace('-', '')

    if not card_number.isdigit():
        return "Invalid"

    for i in range(len(card_number) - 3):
        if card_number[i] == card_number[i + 1] == card_number[i + 2] == card_number[i + 3]:
            return "Invalid"

    return "Valid"
```

```python
N = int(input())

for _ in range(N):
    card_number = input()

    result = is_valid(card_number)
    if result == "Valid":
        print("Valid")
    else:
        print("Invalid")
```

Test Cases:



21. Words Score

Task and Code:

HackerRank | Prepare > Python > Debugging > Words Score

Exit Full Screen View

Problem

Submissions

Leaderboard

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.

Change Theme Language Python 3

```python
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))
```

Line: 3 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

```
2
hacker book
```

Expected Output Download

```
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 functions 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.

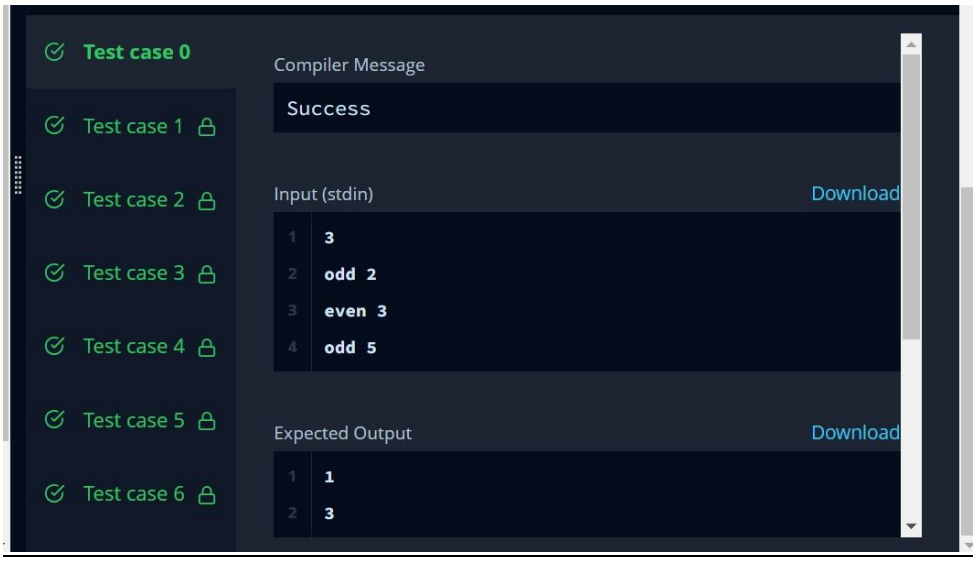
Input Format

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:

```python
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

def print_from_stream(n, stream=None):
    if stream is None:
        stream = EvenStream()

    for _ in range(n):
        print(stream.get_next())

queries = int(input())
for _ in range(queries):
    stream_name, n = input().split()
    n = int(n)
    if stream_name == "even":
        print_from_stream(n)
    else:
        print_from_stream(n, OddStream())
```

Test Cases:



Hard Level Tasks:

23. Maximize It!

Task and Code:

HackerRank | Prepare > Python > Itertools > Maximize It!

Exit Full Screen View

Problem

Submissions

Leaderboard

You are given a function $f(X) = X^2$. You are also given K lists. The ith 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:

$$S = (f(X_1) + f(X_2) + \ldots + f(X_k))\%M$$

X_i denotes the element picked from the ith 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 ith list, followed by N_i space separated integers denoting the elements in the list.

Change Theme Language Python 3

```python
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

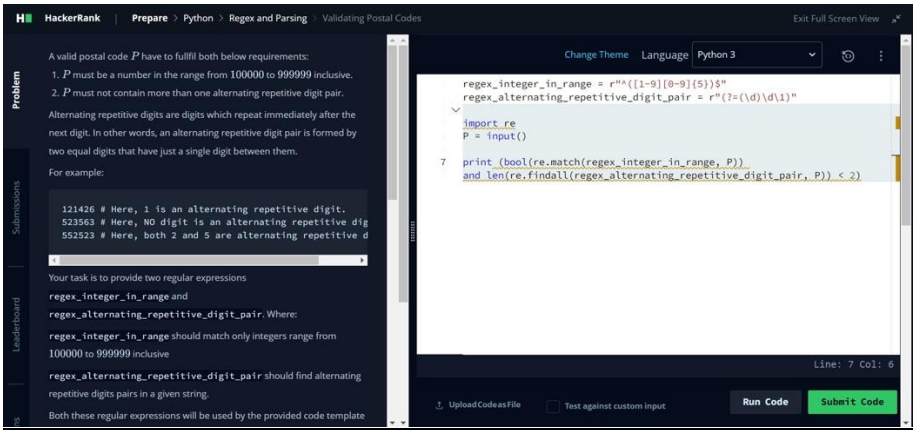
Success

Input (stdin) Download

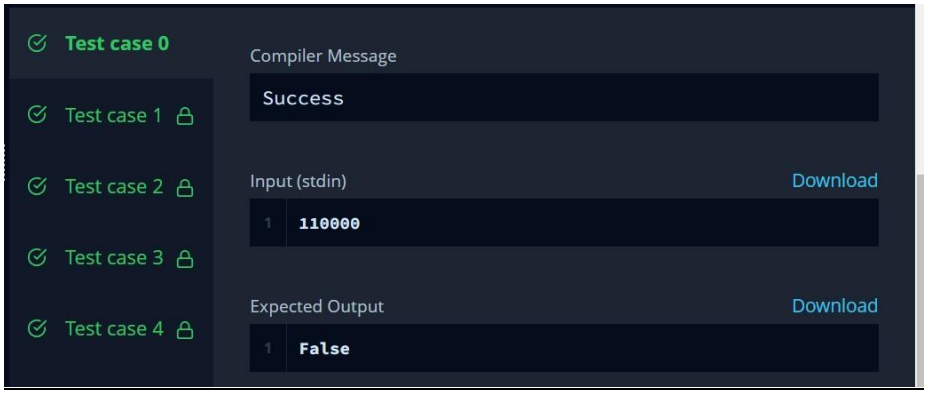
```
7 867
7 6429964 4173738 9941618 2744666 5392018 5813128 9452095
7 6517823 4135421 6418713 9924958 9370532 7940650 2027017
7 1506500 3460933 1550284 3679489 4538773 5216621 5645660
7 7443563 5181142 8804416 8726696 5358847 7155276 4433125
7 2230555 3920370 7851992 1176871 610460 309961
```

24. Validating Postal Codes

Task and Code:

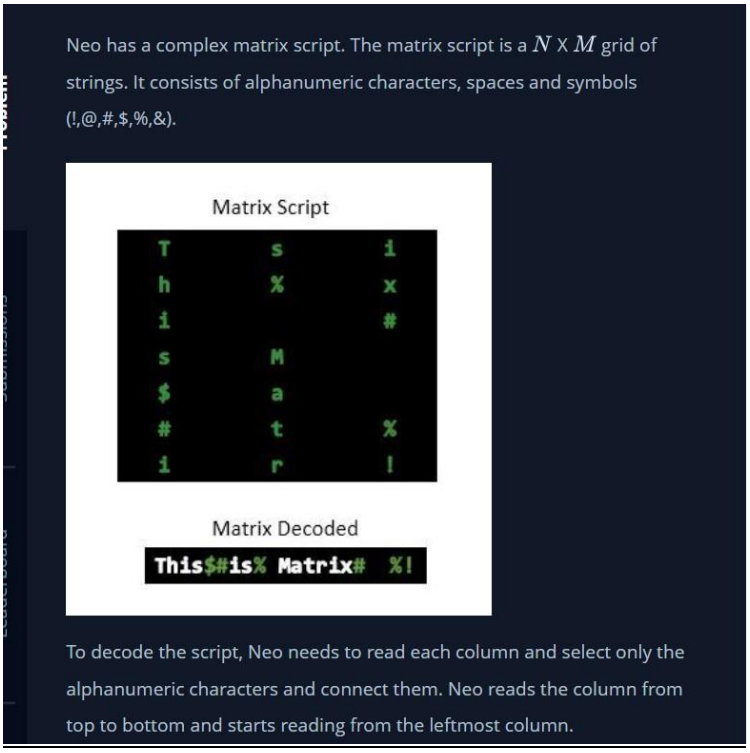


Test Cases:



25. Matrix Script

Task:



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:

```python
import math
import os
import random
import re
import sys
first_multiple_input = input().rstrip().split()
n = int(first_multiple_input[0])
m = int(first_multiple_input[1])
matrix = []
t = []
for i in range(n):
    matrix_item = [x for x in input()]
    matrix.append(matrix_item)
for i in range(m):
    for j in range(n):
        t.append(matrix[j][i])
s = ''.join(t)
path = re.compile(r'\b[ !@#$%&]+\b', re.M)
k = re.sub(path, ' ', s)
print(k)
```

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) Download

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

Expected Output Download

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