KPR Institute of Engineering and Technology
(Autonomous)
Avinashi Road, Arasur, Coimbatore - 641 407

LEARN BEYOND

INFYTQ JAN 2022
QUESTIONS HANDBOOK

Prepared By:
Career Development Center, KPRIET.

SPECIAL NUMBERS

Problem Statement

Given a number innum greater than 100 print outnum as per the below logic:

- Form a collection of 2-digit numbers considering the consecutive digits starting from the leftmost digit in innum.
- For each of the 2-digit numbers in the collection, find the number of special factors
    - Special factor is a factor which is also present in innum.
- Print outnum, the 2-digit number in the collection which has the maximum number of special factors
    - If more than one 2-digit numbers are having maximum number of special factors, consider the largest 2-digit number

Input Format

Read innum from the standard input stream

Output Format

Print outnum to the standard output stream

Sample Input

2340567

Sample Output

56

Explanation

For the given innum, the collection of 2-digit numbers is 23, 34, 40, 05, 56 and 67. The number of special factors for each of the 2-digit numbers are

- For 23, the factors are 1 and 23. Only one factor (23) is present (2340567) in innum. Hence the number of special factors is one
- For 34, the factors are 1, 2, 17 and 34. Two factors (2 34) are present in innum. Hence the number of special factors is two
- For 40, the factors are 1, 2, 4, 5, 8, 10, 20, 40. Four factors (2, 4, 5, 40) are present in Innum. Hence the number of special factors is four
- For 05, the factors are 1 and 5 Only one factor is present in innum Hence the number of special factors is one
- For 56, factors are 1,56,2,28 4,14,8,7 Four factors (2,4,7,56) are present in innum Hence the number of special factors is four
- For 67, factors are 1,67. Only one factor (67) is present in innum. Hence the number special factors is one

  Here 56 and 40 have the maximum number of factors present in innum. Since there are more than one 2-digit numbers having maximum number of special factors, consider the largest one is 56 and hence the Output

Language

Python 3.10

PRIME SUBSTRING

Problem Statement

Consider a positive number innum. Identify and print the number outnum based on the below logic

- Find all the possible sub sequences without leading 0's of the innum which form prime numbers
    - Subsequence: A subsequence of a number num is the possible set of digits formed from num starting from left to right
- Assign the highest identified subsequence to outnum
- If there exists no such subsequence print -1.

Note: 0 and 1 are not prime numbers

Input format

Read the innum from the standard input stream.

Output format

Print the outnum to the standard output stream.

Sample Input 1:

50678

Sample Output 1:

67

Explanation 1:

For the given innum i.e. 50678 the sub sequences without leading 0's formed are 5, 50, 506,5067, 50678, 5068, 507, 5078, 508, 56, 567, 5678, 568, 57,578, 58, 6, 67, 678, 68, 7, 78 and 8. Among these 5, 7 and 67 are prime numbers. Hence 67 is the highest prime number. Hence the output.

Sample Input 2:

140

Sample Output 2:

-1

Explanation 2:

For the given innum, i.e. 140 the subsequence formed are 1, 14, 140, 10, 4, 40 and 0. None of the sub sequences are prime numbers. Hence the output is -1

Language

JAVA

‘M’ LARGEST NUMBERS FROM 2 ARRAYS

Problem Statement

Consider two non-empty positive integer arrays inarr1 and inarr2 as inputs. Identify and print a 2 x m matrix outmatrix based on the logic given below:

- Search common integers between inarr1 and inarr2.
- Select the smallest integer, m, among the common integers. If the smallest integer is not a single digit number, then consecutively add its digits until a single digit, m, is obtained.
- Based on the single digit, m, obtained from above, pick those many largest elements each from inarr1 and inarr2.
- Form outmatrix, in which first row will contain m largest elements selected from inarr1 and second row will contain m largest elements selected from inarr2. Each row must contain elements in ascending order.
- If no common integer exists or smallest common integer value is zero or larger than any of the input array size print output as -1

Input Format

Read the inarr1 from line 1 with array elements separated by (,) comma.
Read the inarr2 from line 2 with array elements separated by (.) comma.
Read the input from the standard input stream

Output format:

Print outmatrix, with each row in newline and row elements are separated by (,) comma, or -1 accordingly.

Print the output to the standard output stream.

Sample Input 1:

0,17,61,65,90
5,0,1,4,100

Sample Output 1:

-1

Explanation 1:

For the given inputs inarr1 is [0,17,61,65,90] and inarr2 is [5,0,1,4,100].

- Smallest common integer is zero.

Output is "-1".

Sample Input 2:

101,101,610,447,389
610,4,101,4,101

Sample Output 2:

447,610
101,610

Explanation 2:

For the given inputs inar1 is [101,101,610,4 47,389] and inarr2 is [610,4,101,4,101]

- Common integers between inarr1 and inam2 are 101, 610.
- Smallest common integer is 101 which is not in single digit Sum of all the digits of 101 is 1+0+1=2 i.e. m=2
- Select two largest elements from inarr1 as "447,610" and from inarr2 "101,610" in ascending order.
- Adding the selected largest elements of inarr1 as first row and the selected largest elements of inarr2 as second row,
- outmatrix is

  447,610
  101,610

Language

JAVA

MAX-REPEATING SUB SEQUENCE AFTER CONCATNATION

Problem Statement

Consider three non-empty arrays inarr1, inarr2 and inarr3 with non empty strings as elements where each string element consists of only alphabets. Identify and print an array outarr based on below logic:

- For each element, elem, of inarr3, identify count, the number of occurrences of elem as a subsequence in a string str1 where str1 is obtained by concatenating a string of inarr1 with a string of inarr2
    - Subsequence: A subsequence of a string str is the possible set of characters formed from str starting from left to right
- Identify the element of inarr3 with the non-zero highest value of count and add it to outarr.
    - If more than one element has the same highest value count add the elements to outarr in their decreasing order of length and in case of same length, add the elements in alphabetical order
- If there is no element with non-zero count, print -1

Note: Do case-insensitive comparison

Input Format:

First line contains the elements of inarr1 separated by. (comma)
Second line contains the elements of inarr2 separated by."(comma)
Third line contains the elements of inarr3 separated by. (comma)
Read the input from the standard input stream.

Output Format:

Print outarr with the elements separated by. (comma) or-1 accordingly to the standard output stream

Sample Input 1:

Welcome,laughter,water
Hello,saino
Win,lAno,xyz

Sample Output 1:

lAno,win

Explanation 1:

For the given input, inarr1 is ['welcome', laughter', 'water"], inarr2 is ['Hello','saino'] and inarr3 is ['lano','win','xyz']. For each elem of inarr3 the count and the corresponding strings where elem occurs as subsequence are:

- 'win': 'win' occurs as a subsequence of 'welcomesaino' and ‘watersaino’. Hence its count is 2.
- “lAno”: “lAno” occurs as a of ‘welcomesaino' and ‘laughtersaino’. Hence count is 2
- ‘xyz':’ xyz' does not occur as a subsequence in any str1. Hence its count is 0

Here, two inarr3 elements are having the same count (2) so add the string to outarr in decreasing order of length. Hence, outarr is ["lAno","win"]. Hence the output.

Sample Input 2:

Cars,walk,mill
Sent,ward,look
Sary,lock,mine

Sample Output 2:

-1

Explanation:

For the given input, inarr1 is ['Cars', 'walk', 'mill'], inarr2 is ['Sent', 'ward', 'look'] and inarr3 is ['Sary', 'lock', 'mine']

For each elem of inarr3 the count and the corresponding str1 where elem occurs as subsequence are:

- 'Sary': 'Sary' does not occur as a subsequence in any str1. Hence its count is 0
- 'lock': 'lock' does not occur as a subsequence in any str1. Hence its count is 0
- 'mine': 'mine' does not occur as a subsequenc e in any str1. Hence its count is 0.
- Here none of the elements of inarr3 have non zero count.
- Hence the output

Language:

Python3.10

UNIQUE ARMSTRONG NUMBER

Problem Statement

Consider a non-empty array Inarr of unique elements such as satisfies the following condition 0<element<1000 identity and p the below logic:

- Starting from the leftmost element in inarr, for each element
- Form all possible numbers by concatenating pair of elements in inarr such that the second element of the pair appears after the first element in inarr.
- From the numbers formed, find and add the unique Armstrong numbers) outarr in the order of occurrence of elements in inarr.
    - If two pairs have the same first number, then consider the order of occurrence of second elements in the pairs
    - An Armstrong number is equal the sum of its own digits each raised to the power of the number of digits in the number
- If there is no Armstrong number found, print-1 as output

Input Format:

First line contains inarr, with the elements separated by ‘,’ (comma)
Read the inputs from the standard input stream

Output format:

Print outarr, with the elements separated by: (comma) to the standard output stream

Sample Input

15,3,1,70,53,71

Sample Output

153,370,371

Language:

JAVA

8 BIT ENCODING

Problem Statement

Consider a non-empty input string instr. Print the outstr which is the encoded value of instr as per the following rules:

- Input string can have alphabets, digits, '+' and '7. Print. -1 if instr contains any other characters
- Starting from the leftmost character in instr, convert the ASCII value of each character into its 8-bit binary value and form a sequence of 8-bit binary values by joining them in their order of conversion
- Divide the 8-bit sequence formed above into 6-bit binary chunks
- Starting from the leftmost 6-bit chuck convert each 6-bit chunk to a decimal number and map the converted decimal number to a character as per the below character mapping and add them to outstr.
- Character mapping > 0-A, 1-B... 25-Z, 26-a, 27-b., 51-z, 52-0, 53-1,…, 61-9, 62-+, 63-/
- Append 0's to the last chunk if it has less than 6 bits

Input Format:

Read the input string instr from the standard input stream.

Output Format:

Print the encoded string outstr or -1 accordingly to the standard output stream.

Sample Input 1:

ABC123+

Sample Output 1:

QUJDMTIZKw

Explanation 2:

For the given input instr "ABC123+", the 8-bit binary value for the ASCII value of each character, starting from the left most character are as follows!

"A" -> 01000001
"B" -> 01000010
"C" -> 01000011
"1" -> 00110001
"2" -> 00110010
"3" ->00110011
"+" -> 00101011

And the sequence is 010000010100001 001000011001100 010011001000110 01100101011

The 6-bit binary chunks created from the above sequence, their decimal values and the corresponding character mapped are as follows:

010000 -> 16 -> "Q"
010100 -> 20 -> "U"
001001 -> 09 -> "J"
000011 -> 03 -> "D"
001100 -> 12 -> "M"
010011 -> 19 -> "T"
001000 -> 08 -> "I"
110011 -> 51 -> "z"
001010 -> 10 -> "K"
110000 -> 48 -> "w"

Adding the mapped characters to outstr results in "QUJDMTIzKw". Hence the output.

Sample Input 2:

Lane-1

Sample Output 2:

-1

Explanation 2:

For the given input instr "Lane-1", 5th special character character is a which is not allowed. Hence the outstr will be -1

Language:

Python3.10

LARGEST VALUE WITH ALL DIGITS 0-5

Problem Statement

Consider a non-empty string array inarr whose elements contain only digits from 0 to 5 such that no element in inarr will contain all the digits from 0 to 5. Identify and print outstr using the logic given below:

- Identify the non-consecutive unique pairs of elements of inarr, starting from left-most element, such that all the digits from 0 to 5 are present at least once in the pair of elements.
- Form strings for each pair identified by concatenating the elements of the pair such that the order of their occurrence in inarr is preserved.
- Set outstr with the largest string formed in the above step. If more than one string exists as largest strings, set outstr with the string whose first or second element used for concatenation appear first in inarr when traversing from left to right.
    - Largest string is a string with maximum number of characters If no such pair is identified, print -1

Input format:

Read inarr with the elements separated by ''(comma) from the standard input stream

Output format:

Print outstr or -1 accordingly to the standard output stream

Sample Input

30012,250232, 53201,3004355,124111

Sample Output

300123004355

Language

Python3.10

STROBOGRAMMATIC NUMBER

Problem Statement

Consider a non-zero number innum consisting of digits 0,1,6,8,9 ONLY identify and print an array outarr based on the logic given below:

- Identify all possible 4-digit strobogrammatic numbers without leading zeroes that can be formed using the digits of innum any number of times
    - Strobogrammatic number is a number which appears the same when rotated 180 degrees
- Add all the identified 4-digit strobogrammatic numbers to outarr in ascending order.
- If no strobogrammatic number can be formed, print-1

input format

Read the number innum from the standard input stream

Output format

Print outarr with the elements separated by" (comma) or-1 accordingly to the standard output stream

Sample Input

109

Sample Output

1001,

Language:

Python3.10

UNIQUE WORDS FROM A SENTENCE

Problem Statement

Consider a non-empty array inarr containing sentences with words separated by space. Identify and print outarr containing updated sentences of inarr as per the logic below:

- Starting with the first sentence, in each of the sentences remove all the duplicate words while keeping only the first occurrence of the word
    - Duplicate word is identified by comparing all the words across all the sentences
- If a sentence contains only words which are duplicate occurrences and not the first occurrence, then update the sentence with "X" in uppercase
- If all the sentences have unique words across all the sentences, then print -1

Note:

- Perform case-sensitive comparison
- Every sentence in inarr and in outarr would have a single space between the words
- Any sentence, either in inarr or in outarr, would have no trailing or leading spaces

Input format

Read the elements of inarr, with the elements separated by ", from the standard input stream.

Output format:

Print outarr, with the elements occurring in new line, or -1 accordingly to the standard output stream

Sample Input 1

God sees the sees people,god is great great,people sees the god

Sample Output 1

God sees the people
god is great
X

Explanation

For the input inarr, starting with the first sentence the updated sentences are as follows: "God sees the sees people"
Removing the duplicate word "sees" while keeping the first occurrence results in "God sees the people" "god is great great: Removing the duplicate word "great" while keeping the first occurrence results in "god is great" "people sees the god": This sentence contains only words which are duplicate Occurrences and not the first occurrence hence results in "X" Hence the output.

Sample Input 2

All that glitters is not gold

Sample Output 2

-1

Explanation

For the input inarr, all the sentences have unique words across all the sentences. Hence the output is '-1"

Language

JAVA

MAX SIZE RECTANGLE IN MATRIX

Problem Statement

Consider a binary matrix of integers inmatrix of size m X n where m, n > 0. Identify and print an integer outnum based on the logic given below:

- Identify the largest submatrix of inmatrix that contains only 1s and whose size is greater than 1.
- Assign the size of the identified largest submatrix to outnum. The size is calculated as x*y where x represents number of rows and y represents number of cols of the identified largest submatrix
- If no such submatrix exists print -1

Input format:

First line will contain number of rows m of inmatrix
The next m lines will contain the elements of inmatrix. Each line will have n elements separated by ‘,’ (comma)

Sample Input

4
1,1,1,1
1,1,1,1
1,1,1,0
0,1,1,1

Sample Output

9

Language

JAVA

SORTED STRING INDEX DIFFERENCE

Problem Statement

Consider a non-empty input string, instr containing at least two characters which is a combination of lowercase alphabets and digits. Identify and print the output using the below logic:

- Separate the alphabets and digits and sort the alphabets in lexicographical ascending order, sortedalpha, and the digits in ascending order, sorteddigit
- Identify the sort difference for sortedalpha and the sorteddigit
    - Sort difference for a sorted string is the absolute difference between the index of the first occurrence of the smallest element and the index of the first occurrence of the largest element in the original unsorted string, instr
- Form a string by concatenating sortedalpha followed by sort difference of sortedalpha followed by:"(colon) followed by sum of the digits in instr and the sort difference of sorteddigit and assign it to outstr
- If the instr consists of only alphabets or only digits, print -1

Note: The lexicographical ascending order is as follows: a,b,c,.....z

Input format:

Read the instr from the standard input stream

Output format:

Print outstr or -1 accordingly to the standard output stream

Sample Input

gt4r22w7e

Sample Output

egrtw2:153

Explanation

For the given instr, the alphabets in instr in lexicographical ascending order, Sortedalpha, is "egrtw" and the digits in ascending order, sorteddigit is "2247".
The first occurrence of smallest alphabet ‘e’ in instr is present at index 8 and the first occurrence of largest alphabet 'w' in instr is present at index 6 in instr. The absolute difference of these indices is 2. Hence sort difference of sortedalpha is 2
The first occurrence of the smallest digit 2 in instr is present at index 4 and the first occurrence of largest digit 7 in instr is present at index 7. The absolute difference of these indices is 3. Hence the sort difference of sorteddigit is 3.
The sum of these digits is 15.
Concatenation in the required format results in "egrtw2:153" as the output

ALL POSSIBLE PRODUCT OF 'N' FROM AN ARRAY

Problem Statement

Consider an array of non-zero positive integers inarr and a number innum where elements of inarr and innum would be greater than 1. Identify and print a 2d array outarr based on the logic given below:

- Find all possible unique combinations of numbers in inarr where the product of the numbers is equal to innum.
    - The numbers in the combinations must be in ascending order
- A number from inarr may be chosen for the combination unlimited number of times.
- Add the identified combinations to outarr such that the combination with smallest first element should be added first.
    - If more than one combination has the same first element, then the combination having the smallest second element should come first and so on.
- If there exists no such combination, then print -1

Input Format:

First line contains inarr elements separated by comma(,)
Next line contains innum
Read the input from the standard input stream

Output Format:

Print outarr with each combination in a new line, with the elements of the combination separated by ,'(comma) or -1 accordingly to the standard out stream

Sample Input

2,4,6,8
16

Sample Output

2,2,2,2
2,2,4
2,8
4,4

Language:

Python3.10

GAME XO

Problem Statement

Consider a non-empty input array inarr of 13 positive unique integers between 1 and 25 only. Identity and print an integer outnum and a 5 X 6 matrix outmatrix, containing only "X" and "O" (both in uppercase) based on the logic given below:

- The elements in the input array inarr are the positions where X is to be placed in the 5 X 5 outmatrix
- Starting from the first row first column as position 1 and moving column-wise then row-wise up to the last row last column, assign "X" in those positions in outmatrix as given by inarr. Remaining positions in outmatrix are assigned with "O".
- After assigning "X" and "O" to outmatrix,
    - assign 1 to outnum if any tow, column, or diagonal in outmatrix has all the 5 elements as "X"
    - assign 2 to outnum if any row, column or diagonal has all the 5 elements as "O
    - assign 0 to outnum if there is no row, column or diagonal containing only "X" or containing only "0"

Assumption: There will be at only one row or one column or one diagonal which has all 5 elements as "X" or "0"

Input Format:

Read inarr with the elements separated by "(comma) from the standard input stream.

Output Format

First line contains outnum
The next 5 lines contain the rows of outmatrix with the tow elements separated by "."(comma)
Print the output to the standard output stream

Sample input

1,24,20,3,7,8,144,15,25,23,19,22

Sample Output

2
X,O,X,X,O
O,X,X,O,O
O,O,O,X,X
O,O,O,X,X
O,X,X,X,X

Language:

Python3.10

SUM OF INTERVALS IN A SORTED ARRAY

Problem Statement

Consider a non-empty integer array inarr containing at least two integers. Identify and print outarr array, based on the below logic:

- For each of the numbers in inarr, move all the negative numbers to the beginning of the array and positive numbers to the end of the array while maintaining their input order.
- Starting from the first number in the rearranged array, for each pair of adjacent numbers, num1 and num2:
    - find the difference (num2-num1)
    - if the difference is greater than 1, compute the sum of missing numbers between num1 and num2 and store the sum in outarr
    - If the difference is not greater than 1 add 0 to outarr

Note: Consider 0 to be a positive number

Input format:

First line contains inarr with the elements separated by ','(comma)
Read the input from the standard input stream.

Output format:

Print outarr with the elements separated by ',' (comma) to the standard output stream.

Sample Input

-5,5,1,-2,7

Sample Output

-7,9,0,20

Explanation

For the given inarr, move the negative elements to the beginning and positive elements to the end while maintaining the input order results in -5,-2,5,1,7
Starting from the first number in the rearranged array, the pair of adjacent numbers, num1 and num2, their difference and the corresponding value to be added in outarr are:

1. num1= -5, num2= -2 difference = -2- (-5) =3; The difference is greater than 1. Hence, missing numbers are -4 and -3, their sum is -7 and outarr becomes {-7}
2. num1=-2, num2 = 5: difference = 5 -(-2)=7; The difference is greater than 1. Hence, missing numbers are -1,0,1,2,3,4, their sum is 9 and outarr becomes {-7,9}
3. num1= 5, num2 = 1: difference = 1-5= -4; Since the difference is negative add 0 to outarr. Hence outarr becomes {-7,9,0}
4. num1= 1, num2= 7: difference = 7-1 = 6; The difference is greater than 1. Hence, missing numbers are 2,3,4,5,6, their sum is 20 and outarr becomes {-7,9,0,20}

Hence the output

Language

JAVA

FORM NUMBER ONLY ARRAY

Problem Statement

Consider a non-empty 2 X n string matrix inmatrix where 1st row contains string elements with at least one digit and 2nd row contains string elements with only digits. Identify and print a 2 X n matrix outmatrix based on the logic given below

- Starting from the leftmost element in first row of inmatrix, for each element present in first row column col of inmatrix,
- Identify the numberString formed by concatenating the digits of the element in order of their occurrence from left to right and check if the same numberString is present in the second row of inmatrix(in any column)
- If the numberString is present in the second row, add the numberString to the first row of outmatrix at column col and the element without the digits in the second row of outmatrix at column col
- Otherwise, add "NA" (uppercase) in the first row and second row of outmatrix

Assumption: There will be at least one digit and one alphabet present in the elements of first row of inmatrix
Note: col value of first column is 1

Input format:

The first 2 lines will contain the elements of inmatrix. Each line will have n elements separated by ‘,’ (comma).

Sample input

mo3s,123m,4tq5
13,3,45

Sample Output

3,NA,45
mos,NA,tq

Explanation

For the given inmatrix, starting from the leftmost element of the first row in Inmatrix, the element at column col and numberString present are:

- col:1 The element is "mo3s" and its numberString is "3". "3" is present in the second row of inmatrix Hence add "3" to the 1st row of outmatrix and element without the numberString i.e., "mos", to the second row of outmatrix respectively at column i.e.,1
- col:2 The element is "t23m" and its numberString is "23". "23" is not present in the second row of inmatrix. Hence add "NA" in uppercase to the first and second row of outmatrix at column col i.e., 2
- col:3 The element is "4tq5" and its numberString is "45". "45" is present in the second row of inmatrix. Hence add "45" to the 1st row of outmatrix and element without the numberString i.e.,"tq", to the second row of outmatrix respectively at column i.e., 3

Hence the output.

Language

JAVA

MATRIX PERFECT SQUARE

Problem Statement

Consider an integer m X n matrix inmatrix, where m>1 and n>1. Identify and print outnum based on the below logic:

- Traverse the elements of inmatrix in linear way, such that, starting from the first row, the elements are to be traversed from left to right and right to left in alternate row
- Add the elements to temparr in the order of their occurrence during the above traversal until one of the two conditions are met:
    - The sum of elements in temparr is a perfect square
        - A number is a perfect square if its square root is a whole number
    - There are no more elements left for traversal
- Add the perfect squares to sqnumarr
- If there is any element which has not yet been visited in the traversal, repeat the above two steps by reinitializing the temparr with empty array.
- Assign the maximum perfect square in sqnumar to outnum
- If there are no perfect squares identified, print -1

Input format:

First line contains number of rows m of inmatrix
The next m lines contain the elements of inmatrix. Each line will have n elements separated by ‘,’ (comma)
Read the inputs from the standard input stream

Output format:

Print the outnum or -1 accordingly to the standard output stream

Sample Input

3
4,2
3,4
5,6

Sample Output

9

Language

JAVA

UNIQUE STRING COUNT

Problem Statement

Consider a non-empty array of strings, inarr whose elements contain only alphanumeric characters. Identify and print an array of integers outarr based on the below logic:

- Identify unique (case insensitive) string elements in inarr which contain only alphabets and the corresponding count of their occurrences, starting from the leftmost element
- Concatenate the identified counts to outarr while maintaining the order of the first occurrence of the identified unique elements in inarr.
- If there are no elements in inarr which contain only alphabets, print -1.

Note: The elements of inarr would not be empty

Input Format

First line will contain the elements of inarr separated by ','(comma)
Read the inputs from the standard input stream.

Output Format

Print outarr, with the elements separated by ','(comma), or -1 [accordingly to the standard output stream.

Sample Input 1

a123,b456,c789,123

Sample Output 1

-1

Explanation 1

For the given input inarr, there is no element in inarr which contain only alphabets. Hence the output is -1

Sample Input 2

bcdef,abcdefg,bcde,100,BCDEF bcde

Sample Output 2

2,1,2

Explanation 2

For the given input inarr, the case-insensitive unique elements containing only alphabets, starting from leftmost element are:

- "bcdef" which occurs twice ("bcedf" and "BCDEF")
- "abcdefg" which occurs once.
- "bcde" which occurs twice("bcde" and "bcde")

Adding the identified counts in the order of the first occurrence of the unique elements, the outarr is [2,1,2]. Hence the output.

Language:

Python3.10

STRING ROTATE – CONSONANTS AT EVEN INDEX

Problem Statement

Consider a non-empty m X 2 matrix inmatrix where the elements of 1 column are non-empty strings strs and elements of 2nd column are positive integers nums. Identify and print outarr based on the below logic:

- For each str in each row of inmatrix, form a string containing the consonants at the even position after str has been rotated to the right by corresponding num times
- Add the non-empty strings from the above formed strings to outarr in increasing order of their length.
    - If there are more than string with the same length and then consider the alphabetical order while adding them to outarr.
    - If all the strings formed are empty, print -1.

Note:

- Position starts from 1.
- Consider 0 to be a positive number.
- Alphabetical order is A,B,C,D,….. X,Y,Z,a,b,c……x,y,z

Input Format:

First line contest the number of rows m of inmatrix.
The next m lines contain the elements of inmatrix. Each line will have 2 elements separated by ',' (comma).
Read the inputs from the standard input stream.

Output Format

Print outarr with the elements separated by ',' (comma) or -1 accordingly to the standard output stream.

Sample Input

4
Knowledge,3
Education,1
Elephant,5
Building,2

Sample Output

Hn,gl,gkl

Explanation

For the given input, the string formed for each str are as follows

- The first str of the inmatrix is "knowledge" and the corresponding num is 3.
- Rotate str to the right num times i.e., 3, the resultant string is "dgeknowle". The consonants at the even positions are 'g', 'k' and 'l'. Hence the string containing the consonants at the even positions is 'gkl'.

- The second str of the inmatrix is "education" and the corresponding num is 1.
- Rotate str to the right num times i.e., 1, the resultant string is "neducatio". There are no consonants in the even position. Hence the string containing the consonants at the even positions is " " (empty string).
- Similarly for the third and fourth str of the inmatrix the strings containing the consonants at the even positions after rotation are "Hn" and "gl".
- Hence, the adding the non-empty strings "gkl", "Hn" and "gl" in increasing order of their length and then in alphabetical order of their length outarr will be ["Hn", "gl", "gkl"]

Hence the output.

SMALLEST SUBSTRING WITH ALL DISTINCT CHARACTERS

Problem Statement

Consider a non-empty array of strings inarr where each element is a non empty alphanumeric string. Identify and print outstr based on the logic below.

- For each element of inarr, find the first smallest length substring, substr, which contains all the unique(case-sensive) characters of the element.
- From the identified substrs assign the substr with the least length to outstr.
    - If more than one substr has the least length, assign the substr corresponding to the element occurring first in inarr, from left to right, to outstr

Input format:

Read the input inarr with the elements separated by (comma) from the standard input stream

Output format

Print the outStr to the standard output stream

Sample Input

Assign,zzzzzxzsdxs,madammxmsgs,1a12231a

Sample Output:

xzsd

Explanation

For the given inarr, the substr for each of the elements are as follows:

- For the 1st element "Assign", the unique characters present are 'A', 's I, g and 'n'. Hence the smallest substring that contains all these characters will be "Assign".

- For the 2nd element "zzzzzxzsdxs", the unique characters present are 'z', 'x', 's' and 'd'. Here there are two smallest substring that contains all these characters "xzsd" and "zsdx". Since "xzsd" occurs first in the 2nd element, it is considered.

- For the 3rd element "Madammxmsgs" the unique characters present are 'M', 'a', 'd', 'm', 'x', 's' and 'g'. Hence the smallest substring that contains all these characters will be "Madammxms".

- For the 4th element "1a12231a", the unique characters present are '1', 'a', '2', and '3'. Hence the smallest substring that contains all these characters will be "231a"

Out of the identified substr, "xzsd" and "231a" have the least length. Since "xzsd" occurs first in inarr from left to night, the outStr will be "xzsd".

Language:

Python3.10

SORTED ROW SUM OF MATRIX

Problem Statement

Consider a non-empty m X n integer matrix inmatrix and an integer innum where 0<=innum<n. Identify and print a non-empty integer array outarr as per the below logic:

- REARRANGE the rows of inmatrix such that the elements at innum column are arranged in ascending order.
    - If more than one element at innum column are same then the rows with the same element must be placed based on the order of their occurrence in inmatrix while rearranging the rows.
- Identify the row sum of each row of the updated inmatrix.
    - Row sum is the sum of the elements present in a row.
- Starting from the 0th row, add each of the row sums identified for a row in updated inmatrix as rowth element of outarr

Note: row and column would start from 0

Input Format:

First line will contain number of rows m of inmatrix
The next m lines will contain the elements of inmatrix. Each line will have n elements separated by ',' (comma)
The next line will contain innum.
Read the inputs from the standard input stream.

Output format:

Print outarray, with the elements separated by ',' (comma)to the standard output stream.

Sample input

3
12,3,8
8,2,44
6,2,2
1

Sample Output

54,10,23

Explanation

For the given inputs, inmatix is
12,3,8
8,2,44
6,2,2
and innum is 1.
The elements at innum column in each row of inmatrix are 3, 2 and 2.

Rearranging the rows as per elements present at Innum column identified, the updated inmatrix is

8 2 44
6 2 2
12 3 8

Row sum of each row are as follows:

- row=0: The elements of 0 row are 8, 2, 44 and their sum is 54.
- row=1: The elements of row are 6,2,2 and their sum 8
- row=2: The elements of row are 12,3,8 and their sum 23

Hence the output array [54,10,23]

Language:

Python3.10

STRING REPLACE

Problem Statement

Consider a non-empty string instr containing alphabets, digits, special characters and space. Identify and print outstr based on the logic given below:

- Substitute each alphabet in instr with the alphabet that occurs in the same position in the reverse alphabetical order such that every 5^{th} substitution is with an uppercase alphabet and every other substitution is with a lowercase alphabet.
- Substitute each digit with the below symbol in a sequence such that first digit occurring from left in instr is substituted with the first symbol from left in the below sequence, second digit with the second symbol and so on

  _ (underscore), # (hash), @ (at the rate), % (percentage)
- If there are more than four digits in instr, the symbols should be reused in the same sequence.
- Keeping all other characters as-is, assign the updated instr to outstr

Note:

- Alphabetical order(applicable for both lower and upper case)

  a, b, c, d ...,x,y,z
- Reverse alphabetical order(applicable for both lower and upper case)

  z.y,x,w,……c,b,a

Input Format

Read the input string instr from the standard input stream

Output Format

Print the encrypted string outstr to the standard output stream

Sample Input

227b Baker St, London NW3 6XE7

Sample Output

_#@y yzpVi hg, oLmwlm Md% _CV#

Explanation

For the given input insfr substituting alphabets with the alphabet that occurs in the same position in the reverse alphabetical order and every 5 substitution with an uppercase alphabet and every other substitution with a lowercase alphabet, it becomes
227y yzpi hg, oLmwlm Md3 6cv7
Substituting the digits in the updated instr with the symbols #@% in sequence the string becomes _#@y yzpVi hg, oLmwlm Md% _CV#
Keeping all other characters as it is the outstr is _#@y yzpVi hg, oLmwlm Md% _CV# and Hence the output.

Language:

Python3.10

PAIR DIFFERENCE SUM OF AN ARRAY

Problem Statement

Consider a non empty array of positive numbers inarr, identify and print outnum based on the given below:

- For each of the elements in inarr, form group(s) such that in each group
  - at most two parts are formed from the elements of inarr.
  - the elements in the pairs occur in their order of occurrence in inarr.
  - each element in inart occurs in ono pair only and both the elements of the second pair occur after both the elements of the first pair in inarr
- For each of the groups formed above, identity the sum of the difference between the second element and first element of each pair.
- Assign the maximum non-zeto positive sum identified above to outnum.
- If no such pair can be formed of no non-zero positive sum is identified print-1.
-

Note: 0 is a positive number

Input Format

First line will contain the elements of inarr separated by (comma)
Read the input from the standard input stream

Output Format

Print the outnum or accordingly to the standard output stream

Sample Input:

3,3,5,0,0,3,1,4

Sample output:

6

Explanation:

For the given input starting from the first element, some of the groups that can be formed such that the groups have at most two pairs of elements and the elements in the pairs occur in their order of occurrence in inarr with each element in mar occurring in one pair only and both the elements of the second pair occurring after both the elements of the first pair in inarr.

1. Group 1:
   Pair 1:[3,5] -> difference -> 5-3 = 2
   Pair 2:[0,5] -> difference -> 3-0 = 3
   Sum of Difference = 5.

2. Group 2:
   Pair 1:[0,3] -> difference -> 3-0 = 3
   Pair 2:[1,4] -> difference -> 4-1 = 3
   Sum of difference =6

Similarly, many other groups can be formed The maximum non-zero positive sum of the difference is 6

Hence output is 6.

MAKE PERFECT 2 POWER EXPRESSION

Problem Statement

Consider a non-empty array of positive integers inarr. Identify and print outarr based on the below logic:

- For each element in inarr, starting from the left-most element, generate a sequence of numbers such that:
    - Starting number in the sequence is element
    - Next number in the sequence is generated based on the current number
        - If the current number is even, divide it by 2 to get the next number in the sequence
        - If the current number is odd, multiply it by 3 and add 1 to get the next number in the sequence
    - Repeat the above steps and keep generating the numbers until the number 1 is generated
- Identify the count of numbers for each of the sequences generated

Input format:

First line contains the elements of inarr separated by " "(space)
Read the input from standard input stream

Output format:

Print outarr with the elements separated by", to the standard output stream

Sample Input

10 100

Sample Output

7,26

Explanation

The elements of the given inarr are 10, 100. The sequence generated for each element starting from the left most element of inarr and their corresponding count are:

- For 10: Starting number of the sequence is 10. Since 10 is even the 2nd number of the sequence is 10/2=5. Since 5,the 2nd number of the sequence, is odd the 3rd number in the sequence is ((5*3)+1)= 16. Similarly by repeating the steps the sequence generated for 10, 5, 16, 8, 4, 2, 1

The count of numbers in the sequence is 7

- For 100: Starting number of the sequence is 100. Since 100 is even the 2nd number of the sequence is 100/2= 50. Since 50, the 2nd number of the sequence, is odd the 3rd number in the sequence is 50/2=25 Similarly by repeating the steps, the sequence generated for element 100 is: 100, 50, 25, 76, 38, 19, 58, 29, 88, 44, 22, 11. 34, 17, 52, 26, 13, 40, 20, 10, 5, 16, 8, 4, 2, 1
- The count of numbers in the sequence is 26
- Add the counts to outarr in order of the 7,26

Language

PYTHON

MAX K SIZE INTEGER

Problem Statement

Consider two integer arrays inarr1 and inarr2 of length P and Q, both consisting of single digit integers [0, 9], and an integer. innum where innum ≤P+Q.

Identify and print an array outarr based on the below logic:

- Pick digits from inarr1 and inarr2 such that
  - the digits picked when concatenated form the maximum possible integer maxint containing innum digits and
  - the digits picked maintain the order of occurrence in their respective arrays.
- Starting from the leftmost digit in maxint, add each of the digits as elements to outarr

Note: inarr1 and inarr2 can contain duplicates

Input format:

The first line will contain the elements of inarr1 separated by ',' (comma)
The second line will contain the elements of inar2 separated

Output format:

Print the outarr with its elements separated by", "(comma) to the standard output stream.

Sample Input

3,8,4,5
9,1,2,6,7
5

Sample Output

9,8,7,4,5

Explanation

For the given inputs, inarr1 is of length 4, inarr2 is of length 5, and the value of innum is 5.
Some of the possible integers containing innum digits i.e., 5, formed by picking digits from inarr1 and inarr2 such that the digits picked maintain the order of occurrence in their respective arrays are shown below:
93845-9 from inarr2 and 3,8,4,5 from inarr1
98465-9,6 from inarr2 and 8,4,5 from inarr1
34967-3,4 from inarr1 and 9,6,7 from inarr2
987459,7 from inarr2 and 8,4,5 from inarr1
The maximum 5 digit number possible from these two arrays is 98745.
Thus, store the digits in the outarr as [9, 8, 7, 4, 5] such that the number 98745 can be formed.
Hence the output.

Language:

Python3.10

INTEGER REPLACE

Problem Statement

Consider a non-empty array of positive integers inarr. Identify and print outnum based on the below logic

- For each element in inarr that contains at least one 6, replace all 6s present with 9 and reverse the element.
- Find the average of elements in the updated inarr rounded to two decimal places and assign it to outnum

Input Format:

First line contains the array of integers inarr, with the elements separated by ‘,’ (comma).
Read the inputs from the standard input stream.

Output Format:

Print outnum to the standard output stream

Sample Input

43,60,225,76

Sample Output

116.0

Explanation

For the given inarr, the elements with at least one 6 in them are 66 and 76. After replacing the 6s in these elements with 9 and reversing them, the updated numbers are
66-99
76-97
The updated inarr is {43, 99, 225, 97}. The average of the elements in update inar rounded to two decimal places is
(43+99+225+9734) = 464/4 = 116.0. Hence the output.

Language

JAVA

MAXIMUM SUM OF CYCLIC ARRAY

Problem Statement

Given a non-empty array of integers inarr. Identity and print an integer outnum based on the below logic:

- Considering inarr to be a circular array, identity subarray(s) formed with contiguously placed elements in inarr which leads to the maximum sum of elements
- Assign the maximum sum to outnum

Input format

First line contains the elements of inarr separated by ',' (comma)
Read the input from the standard input stream.

Output format

Print outnum to the standard output stream

Sample Input

8,-8,9,-9,10,-11,12

Sample Output

22

Explanation

For the given inarr, considering inarr to be a circular array, some of the subarrays that can be formed with contiguously placed elements in inarr and their sum are
8,-8,9,-9,10 sum =10
8,-8,9 sum=9
10 sum= 10
10,-11,12,8 sum=19
-8,9 sum=1
12,8 sum = 20
12,8,-8,9,-9.10 sum = 22
8-8.09.10-11.12 → sum=11
Here the subarray [12,8, 8, 9,-9] has the maximum sum i.e., 22 Hence the output 22.

Language

JAVA

MINIMUM AND MAXIMUM SUM OF 2D ARRAY

Problem Statement

Consider the following inputs:

- a square matrix, inmatrix containing at least one non-zero value and inmatrix of size n * n where n>0
- an Integer value, innum where 1 <=innum <= n.

identity and print a string outstr based on the below logic

- Identify the minimum non-zero sum, minsum, and the maximum sum, among the sum of elements of all sub-square matrices of size innum x innum
- Form a string using the identified minsum and maxsum in the format minsum:maxsum to outstr
- If the value of minsum and maxsum are equal, then print minsum

Input format:

First line will contain the size n of the inmatrix,
Second line will contain the size of the sub matrices, innum
The next n lines will each contain the n elements of inmatrix separated by a ‘,’ (comma)
Read the input from the standard input stream

Output format:

Print outstr to the standard output stream

Sample Input

4
2
0,0,1,4
0,0,2,5
7,2,5,6
4,1,9,0

Sample Output

3:20

Explanation

For the given inputs, innum is 2. The sub matrices of size 2x2 formed from the given Inmatrix are as below
The sum of the elements of the sub-matrix
(0,0,0,0) is 0
The sum of the elements at the sub-matt
(0,1,0,2) is 3
Similarly, the sum of other sub matrices obtained are 12, 9, 9, 18. 14, 17, 20
Among these, the minimum non-zero sum, minsum is 3 and the maximum sum, maxsum is 20
Hence "3:20" will be the output.

Language

JAVA

STRING MULTIPLY, REPLACE AND CONCAT.

Problem Statement

Consider the following inputs:

- a non-empty array of strings inan such that each string will contain only lower-case alphabets
- a m x 2 matrix Inmatrix such that:
    - the first column contains alphabetical strings, and the second column contains integer.
    - the alphabetical strings in each row will contain only lower-case alphabets.
    - the length of the alphabetical strings in each row will be greater than or equal to the absolute value of the integer.

The elements of inarr have one to one correspondence with the rows of inmatrix identity and print outarr based on the below logic:

- For each row in Inmatrix, based on the integer intval in the respective row replace the characters in alphabetical string of inmatrix with the characters of the corresponding string in inarr.
    - If the integer intval is positive, replace intval number of characters from the left of the alphabetical string in the row with intval number from the left of the corresponding string in inarr
        - If the length of the corresponding string str in Inarr is lesser than the corresponding integer in the row of inmatrix, make the length of at equal to the integer s make its length equal, keep appending the same string str to the original string repeatedly and while appending for the last time, if required truncate the last set of characters
    - Otherwise, consider the absolute value of the integer intval and replace intval number of characters from the right of the alphabetical string in the row with the intval number of characters from the right of the corresponding string in inarr.
- Add the unique replaced strings to outart in decreasing order of their length. If more than one replaced string has the same length, add them in the alphabetical order.

Note:

- Alphabetical order is: a,b,c,…..x,y,z

Input Format:

First line contains the number of rows m of inmatrix.
The next m lines will contain the elements of inmatrix, Each ane will have 2 elements separated by ‘,’(comma).
Next line contains the elements of inar, separated by separated by comma)
Read the inputs from the standard input stream.

Output Format:

Print outarr to the standard output stream separated by commas in a single line

Sample Input

5
Educat,5
Trav,-3
know,3
train,-2
train,-2
pe,welcome,burger,dance,slice

Sample Output

Pepept,trace,burw,tome

Explanation

For the given inputs, the replaced sing for the alphabetical strings in each row of month corresponding Integer intval and string str in inarr are given below.

- For the first row of inmatrix, the alphabetical string is ‘educat’, intval is 5 and the corresponding string str inarr is 'pe'. The length of 'pe' is less than the absolute value of integer 5 in inmatrix. To make the length equal, repeatedly append str twice to get pepepe Since the length of str greater than 5, truncate the last character we get 'pepep'. As 5 is positive, replace 5 character from the left 'educat’ with the 5 characters from the left of 'pepep' to get 'pepept'

- For the second row of inmatrix, the alphabetical string is trav, intval is-3 and the corresponding string str in inarr is 'welcome'. As -3 is negative replace 3, the absolute value of -3 characters from the right 'trav' with the 3 characters from the right of 'welcome' to get tome.

- Similarly for the remaining words the replaced strings are 'burw', 'trace' and 'trace'

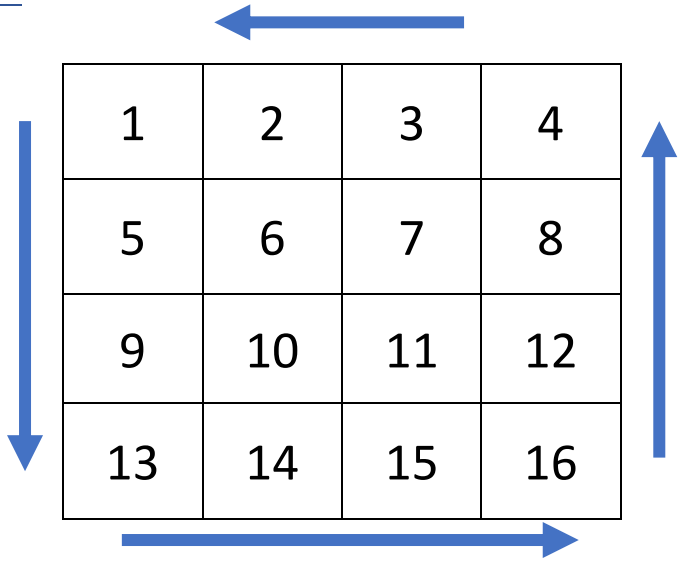
Adding the unique replaced strings in decreasing ceder length and in alphabetical order for the strings with same length outant becomes pepept,trace,burw,tome.
Hence the output.

Language:

Python3.10

AUTOMORPHIC NUMBER

Problem Statement:



Consider a 2D integer matrix inmatrix of size mxn. Identify and print an integer array outarr using the below mentioned logic:

- Traverse through the inmatrix in counterclockwise spiral direction, starting from the element at (0,0)
    - A counterclockwise spiral traversal of a matrix is one where traversal starts with an Element then moves down, then right, then up, then left and so on such that
        - each element is visited only once and
        - the direction will change only when there is no more element left to visit in the current direction.
- During the counter clockwise spiral traversal, keep adding the elements to a temparr in their order of occurrence until the sum of the elements in temparr leads to an automorphic number with length of temparr greater than 1.
- A number, num, is said to be an Automorphic number if and only if its square ends with num.
- Add temparr elements in the order of travesal to outarr.
- Start the travesal again form the next element to be visited, by emptying the temparr.
- Repeat the steps until all the elements have been visited.

Note: There would atleast one temparr whose sum is an automorphic number.

Input format:

First line will contain number of rows m of inmatrix
The next m lines will contain the elements of inmatrix. Each line will have n elements separated by ‘ ’ (space)

Output format:

Print outarray, with the elements separated by ',' (comma)to the standard output stream.

Sample Input:

3
1 2 -2 0

5 -3 2 7
0 1 0 1

Sample Output:

1,5,0,1,0,1,7,0,-2,2,-3,2

Explanation:

For the given array, the counter clockwise traversal is 1,5,0,1,0,1,7,0,-2,2,-3,2. Out of which the formed Automorphic numbers will be

1. [1,5] -> sum = 6 -> 6^2 -> 36
2. [0,1] -> sum = 1 -> 1^2 -> 1
3. [0,1] -> sum = 1 -> 1^2 -> 1
4. [7,0,-2] -> sum = 5 -> 5^2 -> 25
5. [2,-3,2] -> sum = 1 -> 1^2 -> 1

Hence the output.

Language:

Python3.10

STRING PERMUTATION COUNT

Problem Statement

Consider two non-empty arrays inarr1, inarr2 containing only unique single alphabets in lower case as their elements and two non-empty strings instr1, instr2 both containing only lowercase alphabets as inputs.
Identify and print a whole number outnum based on the logic given
below:

- Using the characters from inarr1, generate all possible strings strs such that in each stif every character from inarr1 is used and no character is repeated
- Consider all the generated strings strs, instr1 string and instr2 string. Arrange all of them according to the order of occurrence of their characters in array inarr2
    - To decide the order for any given string, starting from the first character successively check where its character appears in inarr2. If there are multiple strings with the same first character for example, then their second character should be compared with inarr2 and so on
- After arranging all the string, find the number of strings present between instr1 and instr2 and assign it as outnum

Assumptions:
inarr1, instr1 and instr2 will have no character outside of inarr2's characters
instr1 and instr2 will not be present in generated strings strs and they would not be equal

Input format

First line contains elements of inarri as a string separated by ‘,’ (comma)
Second line contains elements of inarr2 as a string separated by ‘,’ (comma)
Third line contains the string instr1
Last time contains the string instr2
Read the input from the standard input stream

Output format

Print outnum to the standard output stream

Sample Input

q,e,u
e,l,q,a,u
equal
queue

Sample Output

3

Explanation

From the given input:
inarr1 = [q,e,u],
inarr2 = [e,l,q,a,u],
instr1= "equal",
instr2 = "queue"

Using all the inarr1 with no characters from repetition, all possible strings strs are "qeu", “que”, “equ”, “euq”, "ueq", "uqe"
On arranging the strings strs, instr1 and instr2 according to the order of occurrence of characters in array inarr2, we get "equ", "equal", "euq", "qeu", "que", "queue", "ueq", “uqe”
There are three strings between instr1 and instr2 hence the outnum is 3

Language

JAVA

MATRIX MULTIPLICATION

Problem Statement

Consider two positive integer matrices inmatrix1, inmatrix2 of size m x n and n x p respectively as inputs. Identify and print an integer array outarr based on the below logic:

- Perform matrix multiplication of inmatrix1 and inmatrix2. Consider the resultant matrix as result_matrix.
- Starting from the first column of result_matrix, from each column select m/2pinteger division) number of elements and add to outrr such that
    - From the first column, the elements from FIRST m/2 rows are selected and from the second column the NEXT m/2 rows are selected and so on in a CYCLIC manner up to the last column resultmatrix
        - Cyclic manner Eg. when selecting elements from a column of resultmatrix, if the last row has been reached then the element from the first row of resultmatrix is considered and continued
- Starting from the first row of resultmatrix, from each row select p2(integer division number of elements and add to outar such that
    - From the first row, the elements from FIRST p/2 cols are selected and from the second row the NEXT m/2 cols are selected and so on in a CYCLIC manner up to the last row of resultmatrix
        - Cyclic manner Eg when selecting elements from a row of resultmatrix, If the last column has been reached then the element from the first column of resultmatrix is considered and continued

Print the elements of outarr with the elements separated by ‘,’(comma)

Assumption 1 <m, n, p<10

Input format:

First line contains number of rows m of inmatrix1
The next m lines will contain the elements of matrix inmatrix1. Each line will have n elements separated by “,”(comma)

The next line contains number of rows n of inmatrix2
The next n lines will contain the elements of matrix inmatri2. Each line will have p elements separated by ‘,’(comma)
Read the input from the standard input stream

Output format:

Print outarr with the elements separated by (comma) to the standard output stream

Sample Input:

```
5
2,8,3,7,5
5,6,6,1,6
7,3,2,6,6
2,4,6,9,4
5,7,7,8,6
5
```

3,7,1,6,5
6,5,6,8,4
4,4,6,8,5
4,4,6,8,1
8,5,7,5,4

Sample Output:

134,127,126,114,179,145,164,160,81,120,134,119,125,164,87,119,114,144,236,120

Explanation:

Perform matrix multiplication and perform matrix traversal as suggested in question.

Language:

Python3.10

LONGEST CONSECUTIVE INTERGER SEQUENCE

Problem Statement

Consider a non-empty array of integers inarr. Identify and print a number outnum based on the logic given below:

- Identify the subsequences of at least two elements in inarr such that elements in the subsequence are consecutive integers.
    - Subsequence: A subsequence of a string str is the possible set of characters formed from str starting from left to right.
- The consecutive integers in the subsequence(s) identified above can be in any position/location in the subsequence.
- From the identified subsequences, assign the length of the longest subsequence to outnum.
- If no such subsequence exists, then print -1.

Input format:

First line would contain the array of integers inarr with the elements separated by ‘,’(comma)
Read the inputs from the standard input stream.

Output format:

Print outnum or -1 accordingly to the standard output stream.

Sample Input 1:

36,41,56,35,44,33,34,92, 43,32,42

Sample Output 1:

5

Explanation 1:

For the given non-empty inarr, there are two subsequences of consecutive elements:

1. 41, 44, 43, 42
2. 2. 36, 35, 33, 34, 32

Of the two, the second subsequence is the longest subsequence of consecutive elements, whose length is 5. Hence outnum is 5.

Sample Input 2:

-1,2,5,7

Sample Output 2:

-1

Explanation:

For the given non-empty inarr, there exists no subsequence of at least two consecutive elements. Hence outnum is-1.

Language:

Python3.10

FIND STRING INDEX OF VOWEL AND CONSONANT

Problem Statement

Consider a non-empty string instr consisting of only lower-case alphabets. Identify and print the string outstr, based on the below logic:

- Identify all the vowels in instr and add them to outstr in lexicographical order
- Considering the first vowel fvowel in outstr, find the index of the first occurrence of fvowel in instr and append it to outstr
    - If there is no vowel in instr add "NA" and append "-1" in place of index
- Identify all the consonants in instr and append them to outstr in lexicographical order
- Considering the last consonant lconsonant in outstr, find the index of the last occurrence of lconsonant in instr and add it to outstr
    - If there is no consonant in instr add "NA" and append"-1" in place of index

Note: Lexicographical order is "abcde....xyz"

Input format:

Read instr from the standard input stream.

Output format:

Print outnum to the standard output stream

Sample Input

temperature

Sample Output

aeeeu6mprrtt7

Explanation

For the given instr, the vowels present in the string are "e", "e", "a", "u", "e". Adding them to outstr based on the lexicographical ordering. outstr becomes "aeeeu".
The first vowel in outstr, fvowel, is "a" and the Index position of the first occurrence "a" in instr i.e., "temperature", is 6. Add the index to outstr to get "aeeeu6".
The consonants present in instr are "t", "m", "p", "r", "t", "r". Adding them to outstr based on the lexicographical order, lconsonant, is “t” and the Index position of the last occurrence “t” in instr i.e., “temperature”, is 7. Add the index to outstr to get "aeeeu6mprrtt7".

Language

JAVA

SPECIAL STRING ARRAY

Problem Statement

Consider a non-empty string array inarr and a string instr. Identify and print outarr, based on the below logic:

- For the string instr, generate a list of special strings specialstringlist.
    - List of special strings A special string for a string stris generated by right shifting the characters by one position. Last character will be shifted as the first character The newly generated special string is used as str for generating the next special string in the list Repeat the generation until the original string stris reached.
    - List of special strings would contain the string str.
- Starting from the leftmost element, for each element in inarr.
    - Add the element of inarr to outarr if it is a subsequence in specialstringlist
        - Subsequence: A subsequence of a string str in the possible set of characters formed from string starting from left to right
- If none of the elements from inarr is a subsequence of specialstringlist print -1

Note: Perform case-insensitive comparison

Input Format

First line contains inarr with the elements separated by ‘,’ (comma)
Second line contains instr.
Read the inputs from the standard input stream

Output format

Print outarr, with the elements separated by ‘,’ (comma) or -1 accordingly to the standard output stream

Sample Input

tarks,arkSt,kStar,trs,tSk
Stark

Sample Output

tarks,arkSt,kStar,trs

Explanation

For the given instr "Stark" the list of special strings specialstringlist generated is:
"Stark", "kStar","rkSta ","arkSt","tarks"

Starting from the leftmost element of inarr, for each element the presence of the element in a subsequence of specialstringlist is given below.

- "tarks" is a subsequence in the special string "tarks". Hence add it to outarr. outarr becomes "tarkS"
- "arkSt" is a subsequence in the special string "arkSt" Hence add it to outarr. Outarr becomes ["tarkS","arkSt"]
- "kStar" is a subsequence in the special string. Hence add it to outarr. outarr becomes ["tarks","arkst, kStar"]
- "trs" is a subsequence in the special string "tarks". Hence add it to outarr. Outarr becomes ["tarks", "arkSt","kStar","trs"]
- "tsk" is not a subsequence in any of the special strings of instr. Hence do not add it to outarr

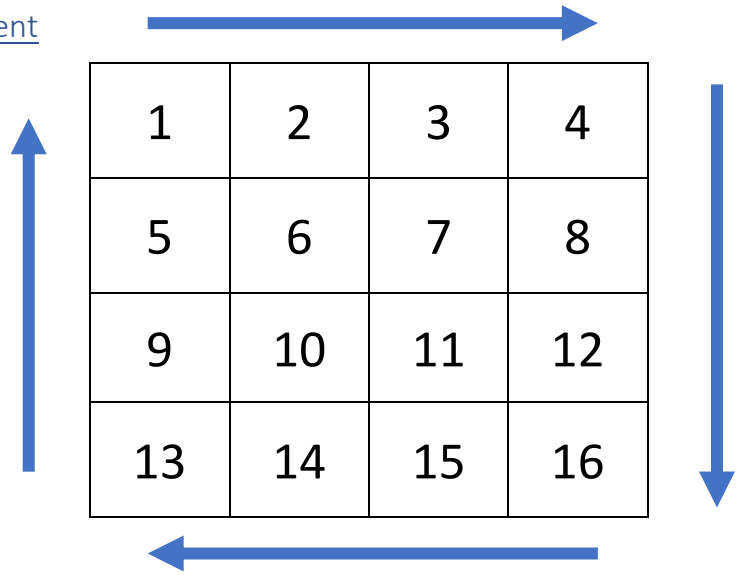
Hence the output.

Language:

Python3.10

SPIRAL MATRIX PALINDROME

Problem Statement



Consider a m x n matrix inmatrix of string elements containing alphanumeric values, where m>1, n >=1. Identify and print outarr based on the logic below:

- Traverse the inmatrix in clockwise spiral way starting from the element at (0,0)
    - A clockwise spiral traversal of a matrix is one which starts at a point then moves right, then down, then left, then up and so on while visiting each element of the matrix exactly once and changing direction only when there are no more elements left to visit in the current direction
- Concatenate the traversed elements as encountered during the clockwise spiral traversal until one of the two conditions is met:
    - The element or the concatenated string is a palindrome with length greater than 1
    - There are no more elements left for traversal
- Add the concatenated string to outarr.
- If there is any element which has not yet been visited in the traversal, repeat the two steps, forming a new concatenated string that starts with the next element present after the last element previously traversed

Note: Perform case-sensitive comparison

Input format:

First line will contain number of rows m of inmatrix.
Read the inputs from the standard input stream.

Output Format

Print outarr, with the elements separated by ‘,’ (comma), or -1 accordingly to the standard output stream.

Sample Input:

3
a3,5,3a
1,35,18

81,52,25

Sample Output:

a353a,18255281,135

Explanation:

For the given array, traverse as instructed in the question and find all possible palindromes.

Language:

Python3.10

UNIQUE SUB STRING

Problem Statement

Consider the following inputs

- a non-empty array of strings inarr containing at least 2 elements, whose elements contain only lowercase alphabets
- two non-zero positive integers innum1 and innum2 such that 1 <= innum1 <= innum2 < length of inarr

Identity and print an integer outnum based on the below logic:

- For each element of inarr, identify the count of the highest occurring character
- Starting from the second element of inarr(from left of inarr), for each element of inarr, identify the character that occurs at the index computed by the modulus of previous element's count and the current element's length
- Generate a string tempstr, by concatenating all the above identified characters
- Identity and assign the count of unique substrings that can be formed from tempstr such that the substrings length is between innum1 and innum2(both inclusive) to outnum

Input format:

First line contains inarr with the elements separated by ‘,’ (comma)
Second line contains innum1
Third and contains innum2
Read the inputs from the standard input stream

Output format:

Print outnum to the standard output stream

Sample Input

trees,of,apple
1
2

Sample Output

3

Explanation

Here, inarr is "trees", "of", "apple", innum1 is 1 and innum2 in 2
For each of the elements in inarr, the highest occurring letter and their number of occurrences are:

- “trees”->’e’, which occurs twice. So, the count for "trees" is 2
- "of"->’o’ & ‘f’, both of which occur once. So, the count for "of" is 1
- "apple"->'p', which occurs twice. So, the count for "apple" is 2

Starting from the second element of inarr, the characters to be considered to generate the new string, tempstr, are:

- 2nd element “of” Here the count of the previous element i.e., “trees” is 2 and length of “of”

- the previous character that is "apple" is 5. Hence the present at the index 1%5= 1, "p", is considered

Hence, by concatenating the characters, tempstr is "op"
The unique substrings of "op" with length between innum1 and innum2, i,e., 1 and 2(both inclusive) are 'o', 'p' and ‘op’
Hence the outnum is 3

Language

JAVA

LARGEST POSSIBLE PALINDROME

Problem Statement

Consider a positive number innum containing at least two digits. Identify and print a number outnum based on the logic given below:

- Identify all possible rearrangements of the digits of innum such that the result is a palindromic number
- Assign the largest palindromic number identified to outnum
- If no palindrome can be formed, print-1

Input format:

Read the innum from the standard input stream

Output format

Print outnum or -1 accordingly to the standard output stream

Sample Input

2341

Sample Output

-1

Explanation

For the given innum, a palindrome cannot be formed by rearranging the digits. Hence the output is -1

Sample Input

388003

Sample Output

830038

Explanation

For the given innum 388003, the following are all rearrangements that result in palindromic number:
380083
308803
803308
830038
From the identified numbers 830038 is the largest. Hence the output is 830038.

Language:

Python3.10

NEXT SMALLEST INTEGER NOT IN ARRAY

Problem Statement

Consider a non-empty array of integers inarr. Identify and print outnum based on the below logic:

- Identify the smallest integer greater than the maximum integer in inarr, which cannot be created by adding any of the integers in inarr as many times as required
- Assign the identified integer to outnum
- If all integers greater than the maximum integer in inarr can be created, print -1.

Input Format:

First line contains inarr with the elements separated by ' '(space)
Read the input from the standard input stream.

Output Format:

Print outnum or -1 accordingly to the standard output stream

Sample Input 1:

2 6 9

Sample Output 1:

-1

Explanation 1:

For the given inarr, the maximum integer in the array is 9.
The smallest integer greater than 9 is 10, which can be created as:6 + 2 + 2 = 10
The next smallest integer is 11, which can be created as: 9+2=11
In this way all integers greater than 9 can be created. Hence the output is -1

Sample Input 2:

4 7

Sample Output 2:

9

Explanation 2:

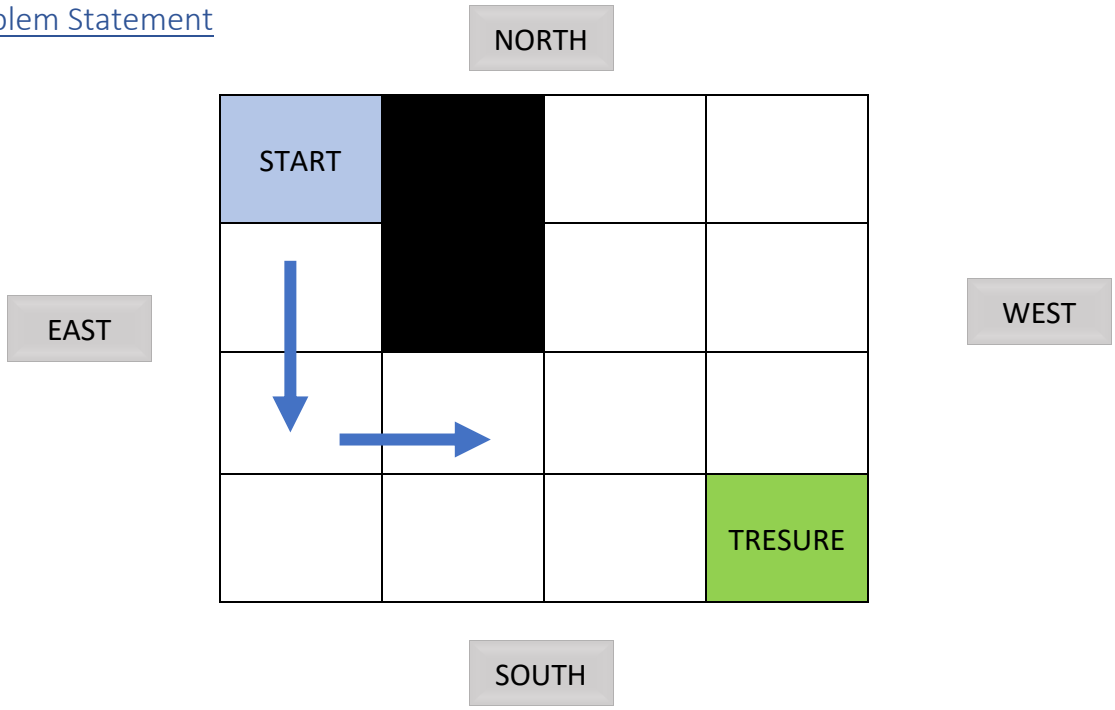
For the given inarr, the maximum integer in the array is 7.
The smallest number greater than 7 is 8, which can be created as : 4+4=8.
The next smallest number is 9, which cannot be formed either using 4 or 7 or both.
Hence the output.

Language:

Python3.10

DIRECTIONS

Problem Statement



Consider the following inputs:

- non-zero positive number innum,
- non-empty array inarr containing 2-digit positive numbers
- non-empty string instr containing only the characters 'N', 'S', 'E', 'W' representing the directions North, South, East and West respectively.

Consider innum x innum square matrix mbx representing a grid having location as 11 for 1st row - 1st column, 12 for 1st row 2nd column up to location as innum-innum (last row - last column)

Identify and print an array outarr using below logic:

- Beginning from the leftmost character of instr traverse through the square matrix mtx starting from the location 11 to reach the last row- last column such that
    - traversal does not happen through the locations mentioned in inarr
    - for any character in instr, if there is no way to move in the direction implied skip that character and choose the next character
- Stop the traversal once the last row last column is reached even if there are more characters left in instr.
- Add the locations visited to outarr as they are encountered in the traversal and print outarr.
- If all the characters of instr have been used for traversal and the last row last column has not been reached print -1

Assumption: The elements of inarr would be valid locations in the square matrix mtx and would not contain 11.

Input format:

First line contains the number innum.
Second line contains the elements of inarr with the elements separated by ‘,’(comma).

Third line contains the string instr.
Read the input from the standard Input stream.

Output format:

Print outarr with the elements separated by ',' (comma) to the standard output stream.

Sample Input:

3
12,22,23
SESEWWEEN

Sample Output:

21,31,32,31,32,33

Explanation:

For the given String "SESEWWEEN" and starting from 11.

1. 'S' -> from 11 move to 21 since 21 is not in array.
2. 'E' -> from 21 we need to move to 22 but not possible since 22 is in array.
3. 'S' -> from 21 move to 31 since 31 is not in array.
4. 'E' -> from 31 move to 32 since 32 is not in array.
5. 'W' -> from 32 move to 31 since 31 is not in array.
6. 'W' -> from 31 move to 30 is not possible since 30 is out of bounds.
7. 'E' -> from 31 move to 32 since 32 is not in array.
8. 'E' -> from 32 move to 33 Hence the destination reached.

Add all the visited indices to the outarr. And Hence the output.

Language:

Python3.10

SORTED PERFECT SQUARE SUB-MATRIX PRODUCT

Problem Statement

Consider a m x n matrix inmatrix containing non-zero positive numbers and a non-zero positive number innum where 1<innum<=m,n. Identify and print outarr based on the logic given below:

- Starting from the element at the first row first column then traversing columnwise followed by rowwise, form all square sub-matrices of size innum x innum possible from inmatrix
- For each submatrix formed, Identify the products of the elements
- Add the identified products to outarr such that all the products which are perfect squares (if any) occur first in outarr, in ascending order, followed by the non perfect square products (if any), in descending order
    - A perfect square is a number whose square root is a whole number(an integer)

Input Format:

- First line contains number of rows m of inmatrix.
- The next m lines will contain the elements of inmatrix. Each line will have n numbers separated by ',' (comma)
- Next line contains the size of square sub matrix innum.
- Read all the inputs from the standard input stream.

Output Format:

Print the outarr with the elements separated by (comma) to the standard output stream

Sample input

3
3,2,6
2,3,4
5,18,10
2

Sample Output

36,144,2160,540

Explanation:

For the given inmatrix and innum, i.e 2, the 2x2 submatrices starting from the first row first column traversing that can be formed columnwise and then rowwise are as follows:
1st Submatrix :
3 2
2 3
Product = 36

2nd Submatrix:
2 6
3 4
Product = 144

3rd Submatrix:
2 3
5 18
Product: 540

4th Submatrix:
3 4
18 10
Product = 2160

Sorted perfect square numbers in ascending order -> 36,144
Add to outarr.

Sorted non-perfect square numbers descending order -> 2160,540
Add to outarr.

Hence the output.

Language:

Python3.10

PERMUTED PRIME

Problem Statement

Given a non-empty string instr consisting of a maximum of 4 unique upper case alphabets, identify and print an array outarr based on the logic given below:

- Determine the ASCII value of each of the alphabets in instr
- Identify all possible combinations formed by concatenating exactly 3 ASCII values by considering the repetition of existing ASCII value(s) for combination
- From the above identified combinations, find the combination(s) which is/are prime number(s).
- For each of the prime numbered combination found, form a 3-letter string obtained by converting the ASCII values back to upper case alphabets.
- Add the 3-lettered string formed to outarr, while maintaining the ascending order of the prime numbered combination(s)
- If a prime number cannot be formed from the ASCII values of the characters, then print -1

Input format:

Read Instr from the standard input stream.

Output format:

Print the outarr with the elements separated by "(comma) or -1 respectively to the standard output stream.

Sample Input

YDE

Sample Output

DEE,DEY,DYE,EDY,EEY

Explanation

For the given instr YDE, the ASCII values of each character are as follows "Y" – 89, "D" – 68, "E" – 69.

The possible combinations formed by concatenating
exactly 3 ASCII values by considering repetition the of ASCII values are:

898989, 686868,696969.
898968, 898969,896889.
896989, 896869,896868.
896968, 896969,686869.
686889, 686968,686869
686989, 688968,688969.
688989, 696989,696968.
696869, 696868,696889.
698969, 698989, 698968.

From the above identified combinations 688969,686989, 686969, 696889 and 696989 are prime numbers. The 3-letter string obtained by converting the ASCII numbers back to upper case alphabets are:
“DEE”, “DEY”, “DYE”, “EDY”, “EEY”. Hence the output.

Language:

Python3.10

ROTATED FACTORS

Problem Statement

Consider a number innum where innum >0. Identify and print the elements of array outarr by using the logic given below:

- Identify set of special numbers for innum
- Set of special numbers for a given number num are (1st digit in num, 1st and 2nd digits in num, 1st 2nd and 3rd digits in num.....)

i.e. starting from the first digit, the other numbers are formed by successively including its next digit

- From the identified special numbers, find all numbers which can divide innum without any remainder and store the unique numbers in ascending order in outarr
- Repeat the above steps by updating innum by shifting its first digit to the end, until innum becomes the original string

Input format:

First line contains string innum
Read the input from the standard input stream

Output format:

Print elements of outarr separated by a ","(comma) to the standard output stream.

Sample Input:

123

Sample Output:

1,3,123,231,312

Explanation:

From the given input, innum is 123.
The set of special numbers for innum are: 1, 12, 123
From the identified special numbers, the numbers that can divide innum without any remainder are 1 and 123.
Hence added to the outarr. The outarr will be [1,123]
Now on shifting innum's first digit to the end, the updated value of innum will be 2311. Again, the set of special numbers for the updated innum are 2,23,231.
From the identified special numbers above, the numbers that can divide the updated innum i.e. 231 without any reminder is 231 only.
Hence, adding 231 to the outarr in ascending order, outarr = [1,123,231].
Now on shifting innum's first digit to the end, the updated value of innum will be 312. Again, the set of special numbers for the updated innum are 3,31,312.
From the identified special numbers above, the numbers that can divide the updated innum i.e. 312 without any reminder ae 3 and 312.
Hence, adding 3 and 312 to the outarr in ascending order, outarr = [1,3,123,231,312].

Language:

Python3.10

REPLACE VOWELS

Problem Statement

Consider a non-empty string instr containing words of alphabets separated by space. Identify and print outarr based on the below logic:

- For each word in instr, form lowercase words by replacing all uppercase alphabets with the corresponding lowercase alphabets
- In each of the lowercase words formed, replace: o each vowel at odd index ind with the vowel present at position ind%5 in sortedvowelarr and each vowel at even index with the position of the vowel in sortedvowelarr sortedvowelarr contains a,e,i,o,u as elements with position of a as 1, position of e is 2, and so on.
- Add the updated words to outarr while maintaining their order of occurrence in instr

Note:
Index starts from 0. Position starts from 1
Consider 0 to be even number

Input Format:

First line contains instr with the words separated by space
Read the input from the standard input stream

Output Format:

Print outarr with the elements separated by (comma) to the standard output stream.

Sample Input

Health is Wealth

Sample Output

ha1lth,3s,wa1lth

Explanation:

For the Example, the words in instr are ‘Health’, ’is’, ‘Wealth’. The lower case words are ‘health’, ‘is’, ‘wealth’. The updated words for each lower case word are as follows:

- ‘health’ -> vowels are present at index 1 and 2. Replacing the vowel at odd index, ind 1 i.e. ‘e’ with the vowel at 1%5=1 in the sortedvowelarr. i.e. ‘a’ and replacing vowel in the even index position 2, i.e. ‘a’ with its position in sortedvowelarr, i.e. ‘1’ the replaced word is ‘ha1lth’
- Repeat the same action for other words as well.

Hence the output ‘ha1lth’,’3s’,’wa1lth’.

Language:

Python3.10

SHRINK THE STRING

Problem Statement

Consider a non-empty string instr containing only lower-case alphabets identify and print outstr based on the logic given below:

- Identify all possible reduced strings of instr
- Reduced string is obtained below.
    - Starting from the left most alphabet in instr, for each adjacent alphabets, alpha1 and alpha2, check it alpha1 occurs immediately before alpha2 in the alphabetical order if yes, then replace both the alphabets alpha1 and alpha2 in instr with a single alphabet which occurs immediately after alpha2 in alphabetical order.
- From the above identified reduced strings, assign the reduced string with the smallest length to outstr
    - If more than one reduced string has the smallest length, assign the reduced string that has the least total sum of the ascii values of its alphabets
- If instr cannot be reduced print -1

Note: Alphabetical order is as shown below
"abcdef……..xyz"
Assumption: The combination "yz"will not occur

Input format

Read instr from the standard input stream.

Output format

Print the string outstr or -1 accordingly to the standard output stream.

Sample Input:

abcegklm

Sample Output:

ahkn

Explanation:

For the given instr, the possible reduced strings that can be formed and the alphabets involved in the reduction are:

- ccegmm : abcegklm -> ccegklm > ccegmm
- ccegkn : abcegklm →> ccegklm ccegkn
- ahmm : abcegklm -> adegklm →> afgklm → ahklm -> ahmm
- ahkn : abcegklm -> adegklm-> afgklm -> ahklm → ahkn

of the four reduced strings. "ahmm" and "ahkn have the smallest length.
Correspondingly the sum of the ascil values for each of the reduced strings are,

- ahmm -> 97+104+109+109 = 419
- ahkn -> 97+104+107+110 = 418

Since "ahkn has the least sum the output is "ahkn".

Language:

Python3.10

STRING VOWEL REPLACE

Problem Statement

Consider a non-empty array, inarr of strings containing only lowercase alphabets and a m x 2 matrix of integers inmatrix such that:

- the size of inarr is equal to m i.e the elements of inarr and the rows of inmatrix have one-to-one correspondence
- each row of integers ‘I’ and ‘j’ of inmatrix would be such that
    - ‘i’ is always less than the length of the corresponding string in inarr
    - ‘j’ would always be less than 26

Identify and print a string outstr based on the below logic:

- For each string str in inarr and corresponding row of integers ‘i’ and ‘j’ of inmatrix, replace all the occurrences of the character at ‘i’th index in str with the ‘j’th element in the alphabet series
- alphabet series -- 'a' would be the 0^{th} element and ‘z’ would be the 25th element
- Starting from leftmost replaced string, identify the first string with the minimum number of consonants and assign it to outstr

Input Format:

First line contains inarr, with the elements separated by "," (comma)
Second line will contain number of rows m of inmatrix
The next m lines will contain the elements of inmatrix. Each line will have 2 elements separated by space.
Read the inputs from the standard input stream.

Output Format:

Print outstr to the standard output stream

Sample Input

apphe,goava,pomegrana te,pip
4
3 11
1 20
8 0
2 18

Sample Output

guava

Explanation

For the given inputs, the strings in inarr after replacement are given below:

- The first element in inarr is "apphe" and the first row in inmatrix is (3,11) All occurrences of the character at index 3 of the word "apphe" ie., ‘h’, is replaced with the 11^{th} element in the alphabet series i.e.. ‘l’, which would result in "apple"
- The second element in inarr is "goava" and the second row in inmatrix is (1,20). All occurrences of the character at index 1 of “goava” i.e., 'o', is replaced with 20^{th} element in the alphabet series i.e., 'u', which would result in “guava”
- The third element in inarr is "pomegranate" and the third row in inmatrix is (8,0). All occurrences of the character at index 8 of "pomegranate" i.e., 'a', is replaced with the 0^{th} element in the alphabet series i.e., 'a', which would result in "pomegranate"
- The fourth element in inarr is “pip” and the fourth row in inmatrix is (2,19). All occurrences of the character at Index 2 of "pip" i.e., 'p' is replaced with the 19^{th} element in the alphabet series i.e., ‘s’ which would result in “sis”

The number of consonants in each of the replaced elements are:
"apple"--> 3
"guava"--> 2
"pomegranate"-->6
"sis"--> 2
Here the minimum number of consonants is present in the strings "guava" and "sis". Since "guava" is the first element with minimum number consonants that occurs when starting from the left most element of inarr, "guava" is assigned to outstr
Hence the output is “guava”.

Language

JAVA

SUBSTRING CONCATENATION

Problem Statement

Consider two non-empty input strings of alphabets instr1, instr2 and a non-zero positive integer innum. Generate and print the output string outstr based on the below logic:

- Starting from the leftmost alphabet, identify innum number of alphabets from instr1 and instr2
- Add the identified alphabets from instr1 followed by instr2 to outstr
- Repeat the above two steps, identifying and the subsequent innum number of alphabets from instr1 and instr2 until either instr1 or instr2 or both have less than innum characters left for processing
- Add the remaining alphabets (if any) in instr1 followed by remaining alphabets (if any) in instr2 to the end of outstr
- If innum is greater than the total number of alphabets in any of the input strings, then take both input strings completely and process in the same order.

Input Format:

First line contains instr1.
Second line contains instr2.
Third line contains innum.
Read the input from the standard input stream.

Output format:

Print the output outstr to the standard output stream.

Sample Input 1

Industry
Ready
7

Sample Output 1

IndustryReady

Explanation

For the given inputs, instr1 is ‘Industry’, instr2 is 'Ready' and Innum is 7.
innum is greater than total number of characters in instr2 i.e., 5.
Hence process instr1 and instr2 in order by adding instrt completely to outstr followed by instr2 completely to outstr resulting in ‘IndustryReady’.
Hence the output

Sample Input 2

Raines
Bowls
2

Sample Output 2

RaBoinwless

Explanation

For the given inputs, instr1 is ‘Raines’, instr2 is ‘Bowls’ and innum is 2.
innum is 2 so starting from leftmost alphabet, take two alphabets, ‘Ra’ from instr1 and 'Bo' from instr2 respectively.
Add the alphabets taken instr1 followed by instr2 to outstr resulting in ‘RaBo’.
As there are more than innum characters left in both instr1 and instr2 left for processing, take the subsequent two alphabets, 'in' from instr1 and ‘wl’ from instr2 respectively.
Add the alphabets taken instr1 followed by instr2 to outstr resulting in “RaBoinwl”.
Since instr2 has less than innum charcters left, add the remaining alphabets ‘es’ from instr1 followed by the remaining character ‘s’ from instr2 to the end of outstr resulting in ‘RaBoinwless’.
Hence the output

Language

JAVA

STRING ARRAY COMPARE WITH CHARACTER ARRAY

Problem Statement

Consider a non-empty string instr containing only lower-case alphabets and an array of strings inarr, containing a collection of words, with its elements separated by (,) comma. There is one-to-one correspondence between the characters of instr and the words in inarr.
Identify and print outstr based on the below logic:

- For each of the unique characters in instr:
    - Identify the index firstind of its first occurrence in instr and the list of indices indexlist for the other occurrences, if any
    - Identify the corresponding string str in inarr at index firstind
    - If the string str is present in inarr across all the identified indices of indexlist and ONLY at these identified indices then the unique character in instr is said to have a special mapped word in inarr
- Add ONLY those unique character(s) having a special mapped word to outstr in their order of occurrence in instr
- If there is no character identified having a special mapped word, assign "NA" in upper case to outstr

Input format:

First line contains instr
Second line contains inarr with its elements separated by (,) comma
Read the inputs from standard input stream

Output format:

Print the outstr to the standard output stream.

Sample Input

dbabbca
bat,ball,rat,bat,bat,car,rat

Sample Output

ac

Explanation

For the given input string instr, the unique characters based on their order of occurrence in instr are 'd','b', 'a', 'c'
For each unique character the firstind, indexlist and corresponding str are:
‘d’→ firstind = 0, indexlist = empty, str="bat"
‘b’→ firstind = 1, indexlist = 3,4, str ="ball"
‘a’→ firstind = 2, indexlist = 6, str ="rat"
‘c’→ firstind = 5, indexlist empty, str = "car"
The unique characters with special mapped word in inarr are identified as follows:
The indexlist for ‘d’ is empty. But its stri.e.,"bat", is not present in inarr only at the identified indices as it occurs in indices 3 and 4 in inarr. Hence 'd' does not have a special mapped word in inarr

The indexlist for 'b' is 3,4. But its str i.e., "ball", is not present in inarr in all the identified indices. Hence 'b' does not have a special mapped word in inarr
The indexlist for 'a' is 6. Its str i.e., "rat", is present in inarr in all the identified indices and only in these indices. Hence 'a' has a special mapped word in inarr
The indexlist for 'c' is empty. Its str i.e., "car", is present in inarr only in the identified indices. Hence 'c' has a special mapped word in inarr
Adding the identified unique alphabets of instr with special mapped word in inarr to outstr in their order of occurrence in instr, outstr becomes "ac". Hence the output.

Language

JAVA

MINIMUM SWAP REQUIRED

Problem Statement

Consider a non-empty array of strings inarr such that inarr will contain:

- ONLY "A" or "B" and
- at least one "A" and one "B"

Identify and print outnum based on the below logic,

- Identify the minimum number of swaps required such that all "A"s are adjacent to each other.
- Assign outnum with the minimum number of swaps
- Print -1 if no swaps are required.

Input format:

Read inarr with the elements separated by, (comma) from the standard input stream.

Output format:

Print outnum, the minimum number of swaps required for the desired arrangement to the standard output stream.

Sample Input

A,B,A,B,A

Sample Output

1

Explanation

For the given input inarr, considering the index of the first letter as 0

- Swapping 'B' at index 1 with 'A' at index results in B,A,A,B,A and then swapping 'B at index 3 with 'A' at index 4 results in B,A,A,A,B
  OR
- Swapping 'B' at index 3 with 'A' at index 0 results in B,B,A,A,A

Considering the above scenarios the minimum number of swaps required is 1. Hence, outnum will be 1.

Language

JAVA

KEY STRING INDEX LIST

Problem Statement:

Consider the following inputs:

- inrow, incol: two non-zero positive numbers
- inkey: non-empty string containing only uppercase alphabets
- inarr1: string elements containing only uppercase alphabets.
- inarr2: unique elements containing exactly 2 digits with the first digit ranging from 1 to inrow and second digit ranging from 1 to incol.

Elements of inarri and elements of inarr2 have one-to-one correspondence
Identify and print a string outstr based on the below logic:

- Form an inrow x incol matrix strmatrix with the elements of inarri such that the elements are placed as mentioned by the corresponding 2-digits in inarr2, where the first digit represents the row, and the second digit represents the column
- Starting from 1st row 1st col and moving columnwise and then rowwise, every element in strmatrix is assigned a location in sequence starting from 1
- For each character in inkey, starting from 1st row 1st col and moving columnwise followed by rowwise:
    - Identify the first element in strmatrix which contains the character
    - Concatenate the corresponding location and character's position in the element identified above and add it to outstr
    - If any of the character is not present in strmatrix then assign "*" for location and position each and add it to outstr.

Note: Position, row and column start from 1.

Input format:

First line contains inarr1 with its elements separated by ',' (comma).
Second line contains inrow.
Third line contains incol.
Fourth line contains inarr2 with its elements separated by (comma).
Fifth line contains inkey.

Read the inputs from the standard input stream

Output Format:

Print the outstr to the standard output stream.

Sample Input:

```
QW,RN,KOI,POL,ERT,XCV,ERB,LHJ,AS
3
3
31,23,21,11,33,13,12,22,32
CARZ
```

Sample Output:

```
328122**
```

Explanation:

For the given inputs, inarr1 is ["QW", "RN", "KOI", "POL". "ERT", "XCV", "ERB", "LHJ", "AS"], inrow is 3, incol is 3, inarr2 is ["31", "23","21","11". "33","13","12","22", "32"] and the inkey is "CARZ".

The matrix strmatrix formed with the elements of inarr1 based on position mentioned by the corresponding 2-digits in inarr2 is:

POL	ERB	XCV
KOI	LHJ	RN
QW	AS	ERT

And their locations are:

(1) POL	(2) ERB	(3) XCV
(4) KOI	(5) LHJ	(6) RN
(7) QW	(8) AS	(9) ERT

Now for each character in instr i.e., CARZ. Finding the first occurrence of each character in strmatrix.

- C -> First found in location (3) in string “XCV” at index (2). Adding location and index to outstr. Hence outstr becomes “32”.
- A -> First found in location (8) in string “AS” at index (1). Adding location and index to outstr. Hence outstr becomes “3281”.
- R -> First found in location (2) in string “ERB” at index (1). Adding location and index to outstr. Hence outstr becomes “328121”.
- Z -> not found in any of the string. Hence add “**” to outstr. Outstr becomes “328121**”.

Hence the output.

Language:

Python3.10

COMMON CHARACTERS

Problem Statement

Consider a non-empty string array inarr where each string contains unique lowercase alphabets of length 1 at least. Identify and print an array outarr using the logic given below.

- Find all possible string pairs from inarr such that the length of second string in the pair is twice the length of the first string
- For each of the above identified pairs, pstr
    - Find the common characters between the string elements of the pair
    - Arrange the common characters in alphabetical order and store it in outarr . If there is no pstr or no common character among any of the pairs pstr, print -1

Note: Alphabetical order is 'a', 'b', 'c', 'd', 'x', 'y', 'z'.

Input format:

Read inarr with its elements separated with ‘,’ (comma) from the standard input stream

Output format

Print the outarr with its elements separated by ‘,’ (comma) or -1 respectively to the standard output stream

Sample Input

acquires,fact,layers,magnitudes,icons,an

Sample Output

a,a,c,i,n,s

Explanation

From the given input inarr, the string pairs pstr where length of second string in a pair is twice the length of the first string are

- "fact" - "acquires" (pstr1)
- "icons- "magnitudes" (pstr2)
- "an"-"fact" (pstr3)

The common characters between strings of pair pstr1 are 'a' and 'c'
Arranging the common characters in alphabetical order and storing in outarr leads to [‘a’,’c’].
Similarly, the common characters between strings of pair pstr2 are ‘n’, ‘i’ and ‘s’. Arranging the common characters in alphabetical order and storing in outarr leads to [‘a’, ‘c’, ‘i’, ‘n’, ‘s’].

Language:

Python3.10