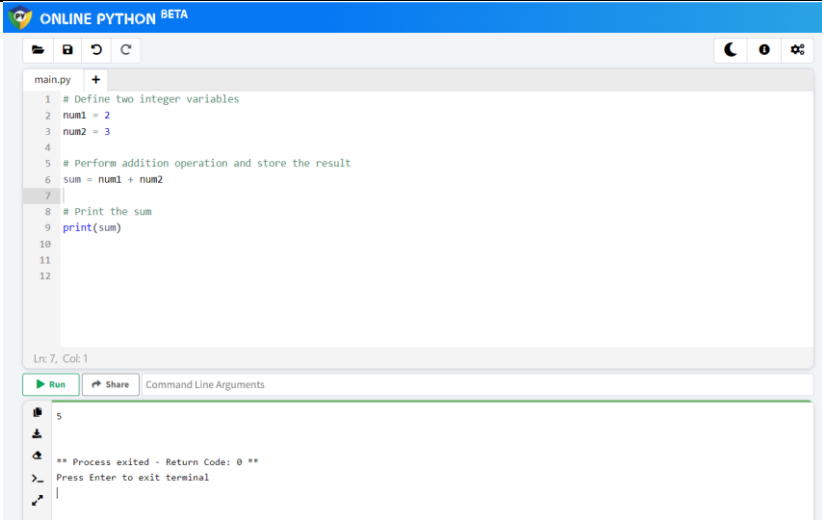


## Koda ģenerācija – Google Gemini

Instruction	Output code	Koda rezultāti
Please generate Python code that defines two integer variables and then does a simple addition operation.	<pre># Define two integer variables num1 = 2 num2 = 3  # Perform addition operation and store the result sum = num1 + num2  # Print the sum print(sum)</pre>	 <p>The screenshot shows an online Python IDE interface. The code editor contains the following Python code:</p> <pre>1 # Define two integer variables 2 num1 = 2 3 num2 = 3 4 5 # Perform addition operation and store the result 6 sum = num1 + num2 7 8 # Print the sum 9 print(sum) 10 11 12</pre> <p>Below the code editor, there are buttons for 'Run', 'Share', and 'Command Line Arguments'. The output area shows the result of the execution:</p> <pre>5  ** Process exited - Return Code: 0 ** Press Enter to exit terminal</pre>

<p>An isogram is a word that has no repeating letters, consecutive or non-consecutive. Implement a function that determines whether a string that contains only letters is an isogram. Assume the empty string is an isogram. Ignore letter case.</p>	<pre>def is_isogram(string):     """     This function determines     whether a string is an     isogram.      Args:         string: The string to check.      Returns:         True if the string is an         isogram, False otherwise.     """     # Convert the string to     lowercase and remove non-     letter characters     clean_string =     ""     for char in string:         if char.isalpha():             clean_string += char.lower()      # Check if there are any     duplicate characters using a     set     return len(set(clean_string))     == len(clean_string)</pre>	<div><div>7 kyu</div><div>Isograms ✓</div><div>☆ 2726 609 90% of 19,826 83,587 of 237,347 chunjef 5 Issues Reported</div><div>Instructions Output Past Solutions</div><div>Time: 488ms Passed: 50 Failed: 0</div><div>Test Results:</div><div>Fixed Tests</div><div>Basic Test Cases (10 of 10 Assertions) Completed in 0.27ms</div><div>Random tests</div><div>Testing for KWEvjzTdNqUJNativW</div><div>Testing for FstXYwWdfTgUboXgMLDSOEbxaQqRzxkeHplniFoyv</div><div>Testing for FICzfscUgmydZX</div><div>Testing for fFEZVDYXKPuWEqzJq</div><div>Testing for hFevEVnkgxWxNOpbERoPDPVWR</div><div>Testing for uQnnPUjQEq</div><div>Testing for gDkRtMqVleyeEnjIUz</div><div>Testing for BXXlHgQXopuxImFCQFVPhPZoQbwYWjHbPMFnm</div><div>Testing for kKypvSXLqGDDBBhKYkdnfbUtYW</div><div>Testing for BjgPUcvXLasoMvc</div><div>Testing for RYkvpm</div><div>Testing for pBQyuJXDJVmRyqpHXmqjrsOsXCnWErsruHYyCKYnSbW</div><div>Testing for CZwBhwCImHZeOcGRN</div></div>
---	--	---

Check to see if a string has the same amount of 'x's and 'o's. The method must return a boolean and be case insensitive. The string can contain any char.

```
def xo(string):  
    """  
    This function checks if a  
    string has the same amount of  
    'x's and 'o's (case-insensitive).  
  
    Args:  
        string: The string to check.  
  
    Returns:  
        True if the string has the  
        same amount of 'x's and 'o's,  
        False otherwise.  
    """  
  
    # Count occurrences of 'x'  
    and 'o' (case-insensitive)  
    x_count = sum(char.lower()  
    == 'x' for char in string)  
    o_count = sum(char.lower()  
    == 'o' for char in string)  
  
    return x_count == o_count
```

7 kyu Exes and Ohs ✓

☆ 1831 ● 396 ↗ 89% of 19,542 ● 82,033 of 235,541 👤 joh\_pot 🚩 2 Issues Reported

Instructions Output Past Solutions

Time: 482ms Passed: 64 Failed: 0

Test Results:

Fixed tests

- > s = 'xo'
- > s = 'XO'
- > s = 'xo0'
- > s = 'xxxoo'
- > s = 'ooxx'
- > s = 'xooxx'
- > s = 'ooxXm'
- > s = 'zpzpzp'
- > s = 'zzoo'
- > s = 'oXoX'
- > s = ''
- > s = 'xxxxxxxxoooo'
- > s = 'xxxxxxxxXooo'
- > s = 'abcdefghijklmnopqrstuvwxyz'

Completed in 0.56ms

Random tests

A pangram is a sentence that contains every single letter of the alphabet at least once. For example, the sentence "The quick brown fox jumps over the lazy dog" is a pangram, because it uses the letters A-Z at least once (case is irrelevant). Given a string, detect whether or not it is a pangram. Return True if it is, False if not. Ignore numbers and punctuation

```
import string

def is_pangram(sentence):
    """
    This function determines
    whether a sentence is a
    pangram.

    Args:
        sentence: The sentence to
        check.

    Returns:
        True if the sentence is a
        pangram, False otherwise.
    """

    # Convert the sentence to
    lowercase and remove non-
    alphanumeric characters
    clean_sentence =
    "".join(char.lower() for char in
    sentence if char.isalnum())

    # Check if all letters from a to
    z are present in the set
    alphabet =
    set(string.ascii_lowercase)
    return
    alphabet.issubset(set(clean_s
    entence))
```

6 kyu

Detect Pangram ✓

☆ 2411 🪙 569 🔄 92% of 12,625 🕒 66,416 of 167,600 👤 anindyabd 🚩 6 Issues Reported

Instructions Output Past Solutions

Time: 457ms Passed: 8 Failed: 0

Test Results:

▼ Fixed tests

> Test pangrams (5 of 5 Assertions)

> Test non-pangrams (3 of 3 Assertions)

Completed in 0.24ms

You have passed all of the tests! :)

Write a function, persistence, that takes in a positive parameter num and returns its multiplicative persistence, which is the number of times you must multiply the digits in num until you reach a single digit.

```
def persistence(num):  
    """  
    This function calculates the  
    multiplicative persistence of a  
    number.  
  
    Args:  
        num: A positive integer.  
  
    Returns:  
        The number of times the  
        digits in num must be  
        multiplied to reach a single  
        digit.  
    """  
    if num < 10:  
        return 0 # Single digit  
        number has 0 persistence  
  
    count = 0  
    while num >= 10:  
        product = 1  
        for digit in str(num):  
            product *= int(digit)  
        num = product  
        count += 1  
    return count
```

6 kyu

Persistent Bugger. ✓

☆ 4293   795   91% of 16,778   66,761 of 185,490   joh\_pot   1 Issue Reported

Instructions   **Output**   Past Solutions

Time: 519ms   Passed: 55   Failed: 0

Test Results:

▼ Persistent Bugger.

> Basic Test Cases (5 of 5 Assertions)

> Random tests (50 of 50 Assertions)

Completed in 1.57ms

You have passed all of the tests! :)

Write a function `cakes()`, which takes the recipe (object) and the available ingredients (also an object) and returns the maximum number of cakes Pete can bake (integer). For simplicity there are no units for the amounts (e.g. 1 lb of flour or 200 g of sugar are simply 1 or 200). Ingredients that are not present in the objects, can be considered as 0.

```
def cakes(recipe, available):  
    """  
    This function calculates the  
    maximum number of cakes  
    Pete can bake based on the  
    recipe and available  
    ingredients.  
  
    Args:  
        recipe: A dictionary  
        containing the ingredients  
        needed for one cake.  
        available: A dictionary  
        containing the available  
        quantities of each ingredient.  
  
    Returns:  
        The maximum number of  
        cakes Pete can bake (integer).  
    """  
  
    # Find the ingredient with  
    the least available quantity  
    relative to the recipe  
    min_ratio = float('inf')  
    for ingredient, amount in  
    recipe.items():  
        if ingredient in available:  
            ratio =  
            available[ingredient] / amount  
        else:  
            ratio = 0  
        min_ratio = min(min_ratio,  
            ratio)  
  
    # Return the floor of the  
    minimum ratio (maximum  
    number of cakes possible)
```

5 kyu

Pete, the baker ✓

☆ 1738 🪙 365 🔄 92% of 5,569 🕒 35,514 of 71,751 👤 BattleRattle 🚩 3 Issues Reported

Instructions Output Past Solutions

Time: 484ms Passed: 56 Failed: 0

Test Results:

static tests

> basic recipes (2 of 2 Assertions)

> missing ingredient

> not enough ingredients

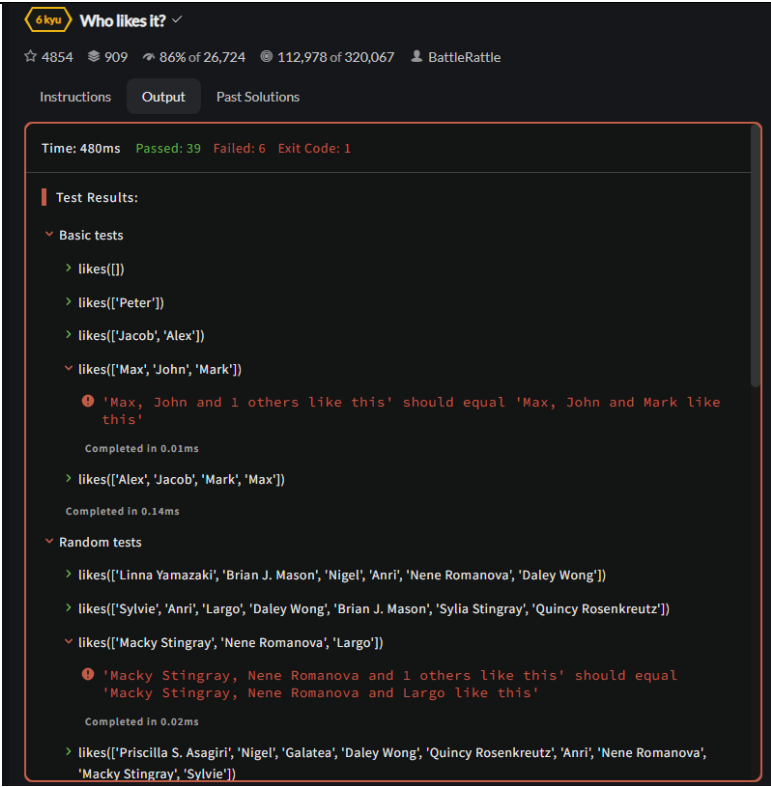
> no ingredients available

> exactly enough ingredients for 1 cake

Completed in 0.24ms

> random tests (50 of 50 Assertions)

You have passed all of the tests! :)

	return int(min_ratio)	
<p>You probably know the "like" system from Facebook and other pages. People can "like" blog posts, pictures or other items. We want to create the text that should be displayed next to such an item.</p> <p>Implement the function which takes an array containing the names of people that like an item. It must return the display text as shown in the examples</p>	<pre>def who_liked_it(names):     """     This function formats the     text displayed based on the     number of people who liked     an item.      Args:         names: A list of names of         people who liked the item.      Returns:         The formatted text to be         displayed.     """     length = len(names)     if length == 0:         return "no one likes this"     elif length == 1:         return f"{names[0]} likes     this"     elif length == 2:         return f"{names[0]} and     {names[1]} like this"     else:         return f"{names[0]},     {names[1]} and {length - 2}     others like this"</pre>	

Digital root is the recursive sum of all the digits in a number. Given  $n$ , take the sum of the digits of  $n$ . If that value has more than one digit, continue reducing in this way until a single-digit number is produced. The input will be a non-negative integer.

```
def digital_root(n):  
    """  
    This function calculates the  
    digital root of a non-negative  
    integer.  
  
    Args:  
        n: A non-negative integer.  
  
    Returns:  
        The digital root of the  
        number.  
    """  
    while n >= 10:  
        sum_of_digits = 0  
        while n > 0:  
            sum_of_digits += n % 10  
            n //= 10  
        n = sum_of_digits  
    return n
```

6 kyu

Sum of Digits / Digital Root ✓

☆ 4344 🏆 829 🔄 90% of 23,772 🕒 90,705 of 258,911 👤 user578387 ⚠️ 2 Issues Reported

Instructions Output Past Solutions

Time: 448ms Passed: 119 Failed: 0

Test Results:

Sum of Digits / Digital Root

> Basic Test Cases (10 of 10 Assertions)

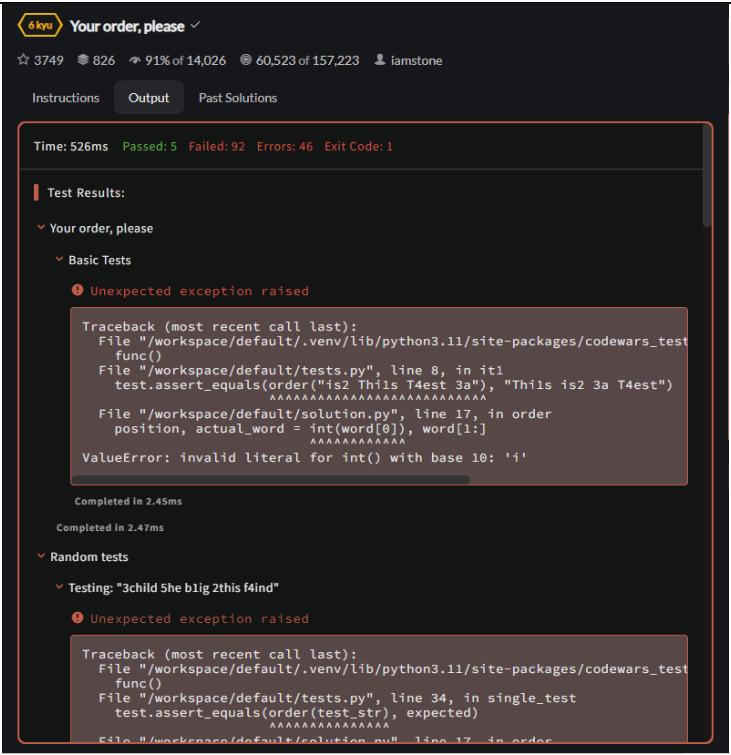
> Single digit tests (9 of 9 Assertions)

> Random tests (100 of 100 Assertions)

Completed in 3.61ms

You have passed all of the tests! :)



<p>Your task is to sort a given string. Each word in the string will contain a single number. This number is the position the word should have in the result.</p> <p>Note: Numbers can be from 1 to 9. So 1 will be the first word (not 0).</p> <p>If the input string is empty, return an empty string. The words in the input String will only contain valid consecutive numbers.</p>	<pre>def sort_by_order(sentence):     """     This function sorts a string     containing words with     position numbers embedded.      Args:         sentence: The string to be         sorted.      Returns:         The sorted string where         each word appears in its         designated position.     """     if not sentence:         return ""      # Split the sentence into     words and extract positions     and actual words     words = []     for word in sentence.split():         position, actual_word = int(word[0]), word[1:]         words.append((position, actual_word))      # Sort the list of tuples based     on the position number     sorted_words = sorted(words)      # Join the sorted words back     into a string     return " ".join([word for _, word in sorted_words])</pre>	
---	---	---

Given an array of ones and zeroes, convert the equivalent binary value to an integer.  
Eg: [0, 0, 0, 1] is treated as 0001 which is the binary representation of 1.

```
def
binary_array_to_number(binary_array):
    """
    This function converts an
    array of ones and zeroes to an
    integer representing the
    binary value.

    Args:
        binary_array: A list
        containing ones and zeroes.

    Returns:
        The integer equivalent of
        the binary value.
    """
    # Reverse the array to
    process bits from left (most
    significant) to right (least
    significant)
    binary_array =
    binary_array[::-1]

    # Initialize the decimal value
    to 0
    decimal_value = 0

    # Iterate through each bit
    (element) in the reversed
    array
    for i, bit in
    enumerate(binary_array):
        # Check if the bit is 1 and
        add the corresponding power
        of 2 to the decimal value
        if bit == 1:
```

7 kyu

Ones and Zeros ✓

☆ 1492 🏆 392 ↗ 90% of 11,145 © 48,924 of 135,408 👤 user4386369 ⚠ 1

Instructions Output Past Solutions

Time: 507ms Passed: 54 Failed: 0

Test Results:

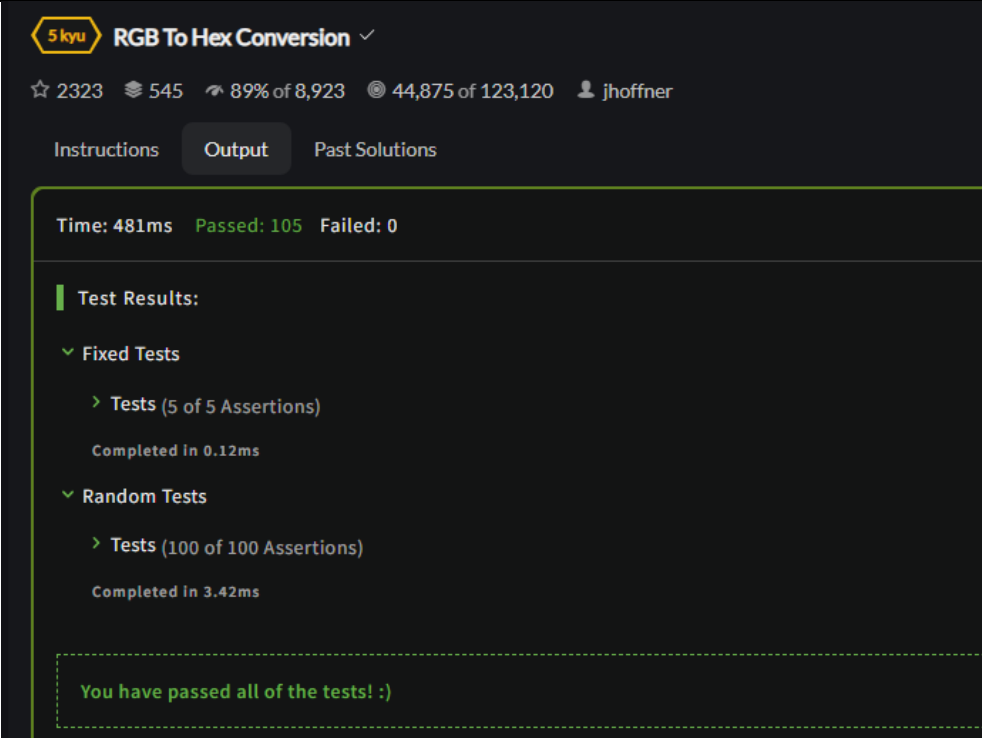
▼ Fixed Tests

> Basic Test Cases (4 of 4 Assertions)

> Random tests (50 of 50 Assertions)

Completed in 1.64ms

You have passed all of the tests! :)

	<pre> decimal_value += 2**i  return decimal_value </pre>	
<p>The rgb function is incomplete. Complete it so that passing in RGB decimal values will result in a hexadecimal representation being returned. Valid decimal values for RGB are 0 - 255. Any values that fall out of that range must be rounded to the closest valid value.</p> <p>Note: Your answer should always be 6 characters long, the shorthand with 3 will not work here.</p>	<pre> def rgb(r, g, b):     """     This function converts RGB     decimal values to a     hexadecimal color     representation.      Args:         r: Red value (0-255).         g: Green value (0-255).         b: Blue value (0-255).      Returns:         A string containing the         hexadecimal color         representation (always 6         characters long).     """      # Clamp RGB values to the     # valid range (0-255)     r = max(0, min(255,     round(r)))     g = max(0, min(255,     round(g)))     b = max(0, min(255,     round(b)))      # Convert each value to     # hexadecimal string with zero-     # padding     hex_r = hex(r)[2:].upper() #     Remove "0x" prefix and     convert to uppercase     hex_g = hex(g)[2:].upper() </pre>	

	<pre>hex_b = hex(b)[2:].upper()  # Ensure each component has two characters (zero-pad if necessary) hex_r = hex_r.zfill(2) hex_g = hex_g.zfill(2) hex_b = hex_b.zfill(2)  # Combine RGB components and return the hexadecimal string return f"{hex_r}{hex_g}{hex_b}"</pre>	
--	--	--

Write an algorithm that takes an array and moves all of the zeros to the end, preserving the order of the other elements.

```
def
move_zeros_two_pointers(arr
):
    """
```

This function moves all zeros in an array to the end using two pointers (in-place).

Args:

arr: A list representing the array.

Returns:

The modified list with zeros at the end.

```
    """
```

# Initialize two pointers: write\_index (to insert non-zero elements) and i (to iterate)

```
    write_index = 0
```

```
    for i in range(len(arr)):
```

```
        # If the current element is not zero, swap it with the element at write_index
```

```
        if arr[i] != 0:
```

```
            arr[write_index], arr[i] = arr[i], arr[write_index]
```

```
            write_index += 1
```

```
    return arr
```

5 kyu Moving Zeros To The End ✓

☆ 2098 🌟 532 🔄 89% of 11,836 🗳 71,854 of 165,967 👤 xcthuhu 🚩 11 Issues Reported

Instructions Output Past Solutions

Time: 503ms Passed: 171 Failed: 0

Test Results:

Fixed Tests

Basic Test Cases (5 of 5 Assertions)

Completed in 0.14ms

Random Tests

- > move\_zeros([9, 8, 5, 1, 8, 2, 3, 1, 2])
- > move\_zeros([0, 0, 0, 0, 0, 0, 0])
- > move\_zeros([])
- > move\_zeros([9, 8, 5, 3])
- > move\_zeros([8, 7, 1, 1, 6, 3, 1, 8])
- > move\_zeros([1, 1, 0])
- > move\_zeros([0, 2, 0, 0, 1, 3, 0, 1, 0])
- > move\_zeros([2, 0, 1, 1])
- > move\_zeros([6, 9, 8, 6, 1, 7])
- > move\_zeros([0, 4, 0, 0, 9, 0, 0, 0, 0])
- > move\_zeros([0, 0, 0, 0])
- > move\_zeros([2, 2, 1, 2])
- > move\_zeros([4, 0, 0, 2, 9, 0, 0, 0, 7, 0, 0])

The goal of this exercise is to convert a string to a new string where each character in the new string is "(" if that character appears only once in the original string, or ")" if that character appears more than once in the original string. Ignore capitalization when determining if a character is a duplicate.

```
def duplicate_encoder(string):  
    """  
    This function encodes a  
    string to indicate unique and  
    duplicate characters.  
  
    Args:  
        string: The string to  
        encode.  
  
    Returns:  
        A new string where "("  
        indicates a unique character  
        and ")" indicates a duplicate.  
    """  
    # Convert the string to  
    lowercase for case-insensitive  
    comparison  
    lower_string = string.lower()  
  
    # Create a dictionary to store  
    character counts  
    char_counts = {}  
    for char in lower_string:  
        char_counts[char] =  
        char_counts.get(char, 0) + 1  
  
    # Build the new string by  
    iterating through the original  
    string  
    result = ""  
    for char in string:  
        if char_counts[char.lower()]  
        == 1:  
            result += "("  
        else:  
            result += ")"
```

6 kyu

Duplicate Encoder ✓

☆ 3923 697 90% of 18,774 78,959 of 206,404 obnounce 3 Issues Reported

Instructions Output Past Solutions

Time: 470ms Passed: 48 Failed: 0

Test Results:

▼ Duplicate Encoder

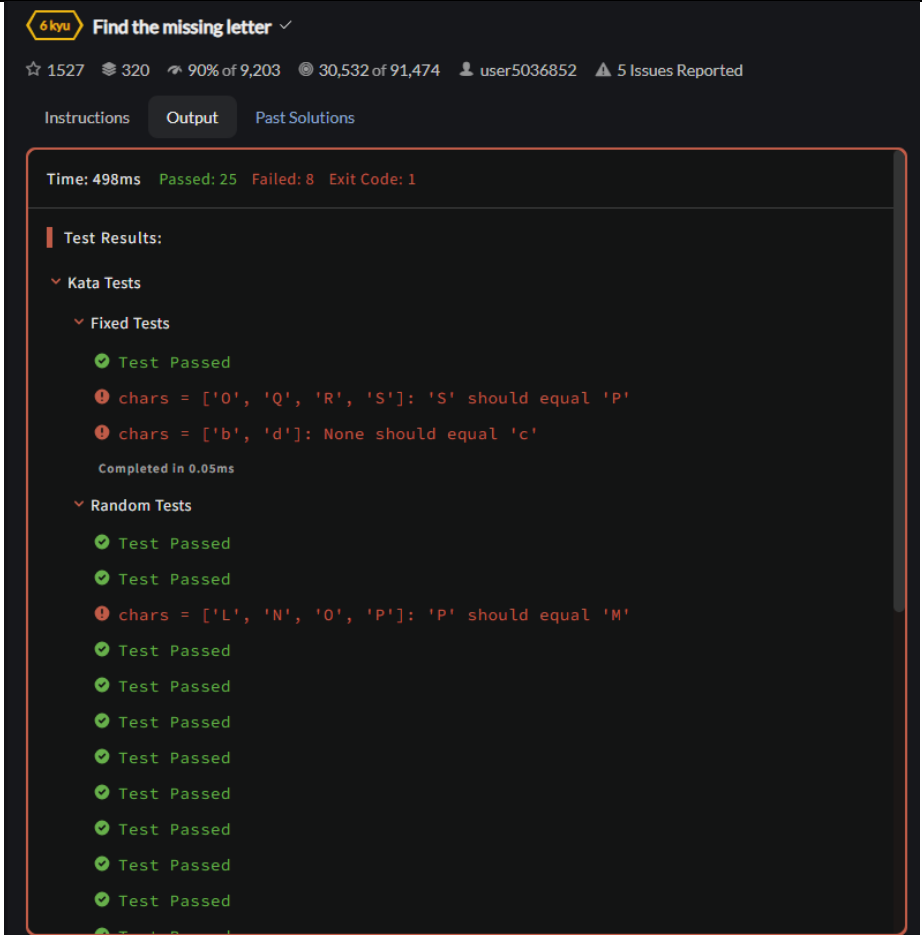
> Basic Test Cases (6 of 6 Assertions)

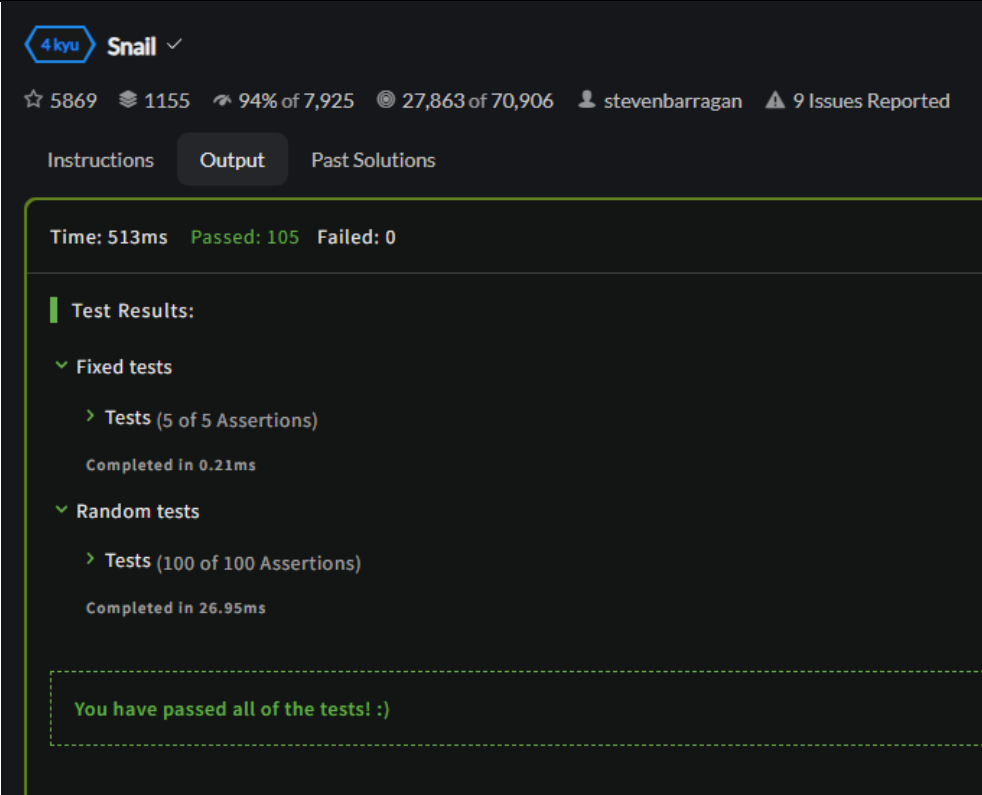
> Tests with '(' and ')' (2 of 2 Assertions)

> And now... some random tests ! (40 of 40 Assertions)

Completed in 5.47ms

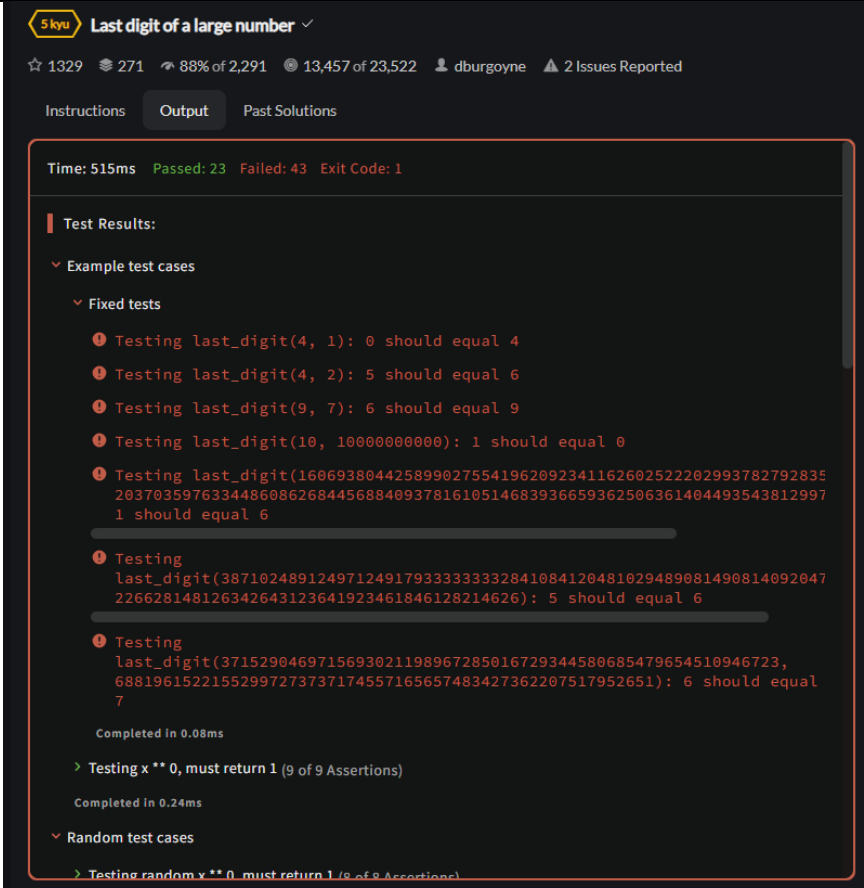
You have passed all of the tests! :)

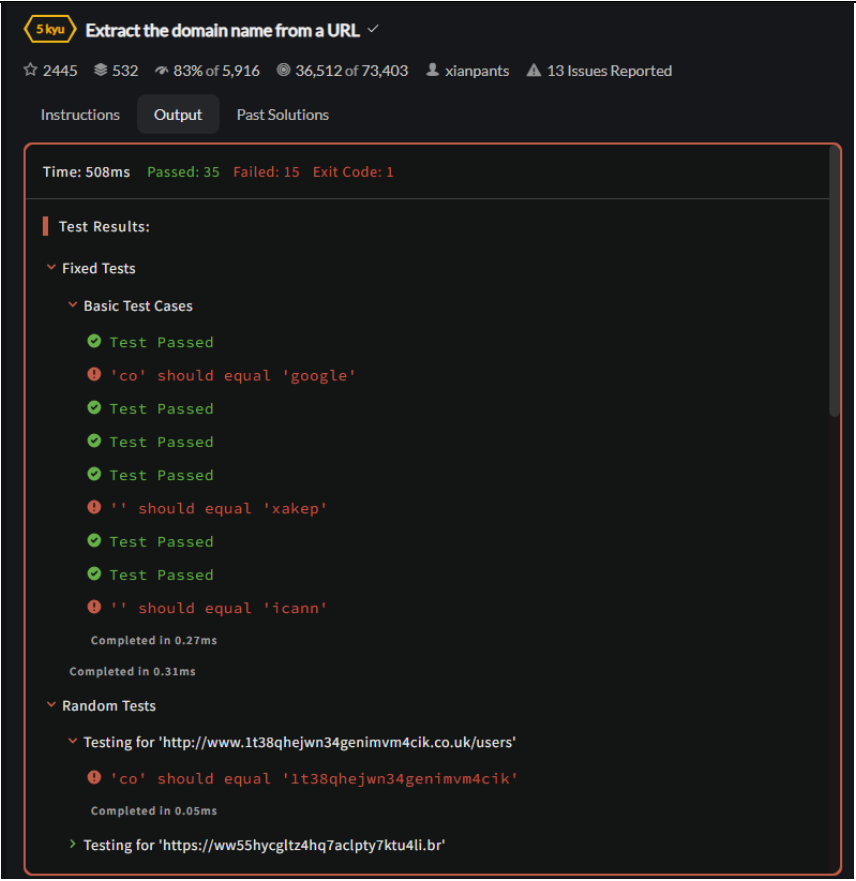
	return result	
<p>Write a method that takes an array of consecutive (increasing) letters as input and that returns the missing letter in the array. You will always get an valid array. And it will be always exactly one letter be missing. The length of the array will always be at least 2. The array will always contain letters in only one case.</p>	<pre>def find_missing_letter(chars):     """     This function finds the     missing letter in an array of     consecutive (increasing)     letters.      Args:         chars: A list of consecutive         (increasing) letters.      Returns:         The missing letter as a         string.     """     # Get the expected     difference between     consecutive letters (always 1)     expected_diff = ord(chars[1])     - ord(chars[0])      # Iterate through the array     from the second element     for i in range(1, len(chars)):         # Check if the current         difference is not equal to the         expected difference         if ord(chars[i]) - ord(chars[i-         1]) != expected_diff:             # Calculate the missing             character's ASCII code based             on the expected difference             missing_char_code =             ord(chars[i-1]) + expected_diff             # Convert the ASCII code             back to a character</pre>	

	<pre> return chr(missing_char_code) </pre>	
<p>Given an n x n array, return the array elements arranged from outermost elements to the middle element, traveling clockwise.</p>	<pre> def spiral_order(matrix):     """     This function returns the     elements of an n x n matrix     arranged in a spiral order     (clockwise).      Args:         matrix: A list of lists         representing the n x n         matrix.      Returns:         A list containing the         elements of the matrix         arranged in a spiral order.     """     result = []     top, bottom, left, right = 0, len(matrix) - 1, 0, len(matrix[0]) - 1     direction = "right"      while top &lt;= bottom and left &lt;= right:         # Move in the current         direction until reaching a         boundary         if direction == "right":             for col in range(left, right + 1): </pre>	



	<pre>result.append(matrix[top][col])     top += 1     elif direction == "bottom":         for row in range(top, bottom + 1):  result.append(matrix[row][right])     right -= 1     elif direction == "left":         for col in range(right, left - 1, -1):  result.append(matrix[bottom][col])     bottom -= 1     elif direction == "top":         for row in range(bottom, top - 1, -1):  result.append(matrix[row][left])     left += 1      # Change direction for the next loop iteration     direction = {         "right": "bottom",         "bottom": "left",         "left": "top",</pre>	
--	---	--

	<pre>"top": "right", }[direction]  return result</pre>	
<p>Define a function that takes in two non-negative integers <math>a</math> and <math>b</math> and returns the last decimal digit of <math>a^b</math>. Note that <math>a</math> and <math>b</math> may be very large!</p> <p>For example, the last decimal digit of 9797 is 99, since <math>97^{97} = 478296997 = 4782969</math>.</p> <p>The last decimal digit of <math>(2200)2300(2200)2300</math>, which has over 10921092 decimal digits, is 66. Also, please take 0000 to be 11.</p> <p>You may assume that the input will always be valid.</p>	<pre>def last_digit(a, b):     """     This function calculates the     last decimal digit of a very     large power a^b.      Args:         a: A non-negative integer         base.         b: A non-negative integer         exponent.      Returns:         The last decimal digit of         a^b.     """     # Handle special cases (0^0     and anything to the power of     0)     if a == 0 and b == 0:         return 1     elif a == 0:         return 0      # Last digit of a can repeat in     a cycle of {1, 0, 5, 6}     last_digit_cycle = [1, 0, 5, 6]     a_last_digit = a % 10 # Get     the last digit of a      # Reduce b's exponent by     finding the remainder when     divided by the cycle length (4)</pre>	 <p>The screenshot shows a coding challenge titled "Last digit of a large number" with a 5 kyu difficulty level. It has 1329 stars, 271 coins, and 88% of 2,291 users solved it. The challenge is by user dburgoyne, with 13,457 of 23,522 users having solved it. There are 2 issues reported.</p> <p>The interface includes tabs for "Instructions", "Output", and "Past Solutions". The "Output" tab is active, showing the following details:</p> <ul style="list-style-type: none"> <li>Time: 515ms</li> <li>Passed: 23</li> <li>Failed: 43</li> <li>Exit Code: 1</li> </ul> <p>The "Test Results" section shows "Example test cases" and "Fixed tests". The tests are as follows:</p> <ul style="list-style-type: none"> <li>Testing last_digit(4, 1): 0 should equal 4</li> <li>Testing last_digit(4, 2): 5 should equal 6</li> <li>Testing last_digit(9, 7): 6 should equal 9</li> <li>Testing last_digit(10, 10000000000): 1 should equal 0</li> <li>Testing last_digit(1606938044258990275541962092341162602522202993782792835203703597633448608626844568840937816105146839366593625063614044935438129971 should equal 6</li> <li>Testing last_digit(387102489124971249179333333332841084120481029489081490814092047226628148126342643123641923461846128214626): 5 should equal 6</li> <li>Testing last_digit(3715290469715693021198967285016729344580685479654510946723, 68819615221552997273737174557165657483427362207517952651): 6 should equal 7</li> </ul> <p>The tests are completed in 0.08ms. Below the fixed tests, there are "Random test cases" with a note: "Testing x ** 0, must return 1 (9 of 9 Assertions)".</p>

	<pre> b_exp = b % 4  # Return the last digit based on the reduced exponent and the cycle return last_digit_cycle[b_exp] </pre>	
<p>Write a function that when given a URL as a string, parses out just the domain name and returns it as a string.</p>	<pre> import urllib.parse  def get_domain_name(url):     """     This function extracts the     domain name from a URL     string.      Args:         url: The URL string.      Returns:         The domain name as a         string, or an empty string if         parsing fails.     """     try:         # Parse the URL using         urllib.parse         parsed_url =         urllib.parse.urlparse(url)         # Extract the netloc         (domain name)         domain_name =         parsed_url.netloc.lower() #         Convert to lowercase         # Split and remove         subdomain (if any)         parts =         domain_name.split(".") </pre>	 <p>The screenshot shows the LeetCode interface for the problem 'Extract the domain name from a URL' (5kyu). It displays statistics: 2445 stars, 532 likes, 83% of 5,916 votes, 36,512 of 73,403 submissions, and 13 issues reported. The 'Output' tab is selected, showing a time of 508ms, 35 passed tests, 15 failed tests, and exit code 1. The test results are categorized into 'Fixed Tests' and 'Random Tests'. Under 'Fixed Tests', there are 'Basic Test Cases' with 6 tests: 4 passed (green checkmarks) and 2 failed (red exclamation marks). The failed tests are: 'co' should equal 'google' and '' should equal 'xakep'. Under 'Random Tests', there is one test case for 'http://www.1t38qhejwn34genimvm4cik.co.uk/users' which failed because 'co' should equal '1t38qhejwn34genimvm4cik'.</p>

	<pre>    return parts[-2] if len(parts) &gt; 1 else domain_name except ValueError:     # Handle potential parsing errors (invalid URL format)     return ""</pre>	
--	---	--

Modern Roman numerals are written by expressing each digit separately starting with the left most digit and skipping any digit with a value of zero. In Roman numerals:

1990 is rendered: 1000=M, 900=CM, 90=XC; resulting in MCMXC

2008 is written as 2000=MM, 8=VIII; or MMVIII

1666 uses each Roman symbol in descending order: MDCLXVI.

Input range :  $1 \leq n < 4000$

In this kata 4 should be represented as IV, NOT as IIII (the "watchmaker's four").

```
roman_numerals_map = {
    "I": 1,
    "V": 5,
    "X": 10,
    "L": 50,
    "C": 100,
    "D": 500,
    "M": 1000,
}

def int_to_roman(num):
    """
    This function converts an
    integer to a Roman numeral
    string.

    Args:
        num: An integer between
        1 and 3999.

    Returns:
        A string representing the
        Roman numeral equivalent of
        the integer.
    """
    result = ""
    for key, value in
sorted(roman_numerals_map.items(), reverse=True):
        while num >= value:
            result += key
            num -= value
    return result
```

Kodu nācās papildināt: pievienot klasi, kas ietver abas šīs funkcijas:

4 kyu

Roman Numerals Helper ✓

☆ 2536 🌐 446 🔗 90% of 2,655 👤 11,161 of 27,209 👤 jhoffner

Instructions Output Past Solutions

Time: 506ms Passed: 307 Failed: 303 Exit Code: 1

Test Results:

tests

fixed tests

to roman

testing to\_roman, for input 1000  
:  
'XX'  
should equal 'M'

testing to\_roman, for input 4  
: 'IIII' should equal 'IV'

Test Passed

testing to\_roman, for input 1990  
:  
'XX'  
should equal 'MCMXC'

testing to\_roman, for input 2008  
:  
'XX'  
should equal 'MMVIII'

Completed In 0.19ms

> from roman (5 of 5 Assertions)

Completed In 0.32ms

random tests

	<pre>def roman_to_int(roman_num):     """      This function converts a     Roman numeral string to an     integer value.      Args:         roman_num: A string         representing a valid Roman         numeral.      Returns:         An integer equivalent to         the Roman numeral.     """      sum = 0     prev = 0     for char in roman_num:         current = roman_numeral_map[char]         # Handle subtractive notation (IV, IX)         if current &lt;= prev:             sum += current         else:             sum = sum - prev + current         prev = current     return sum</pre>	
--	---	--

In this kata we want to convert a string into an integer. The strings simply represent the numbers in words.

Examples:

"one" => 1

"twenty" => 20

"two hundred forty-six" => 246

"seven hundred eighty-three thousand nine hundred and nineteen" => 783919

Additional Notes:

The minimum number is "zero" (inclusively)

The maximum number, which must be supported is 1 million (inclusively)

The "and" in e.g. "one hundred and twenty-four" is optional, in some cases it's present and in others it's not

All tested numbers are valid, you don't need to validate them

```
ones = {
  "zero": 0,
  "one": 1,
  "two": 2,
  "three": 3,
  "four": 4,
  "five": 5,
  "six": 6,
  "seven": 7,
  "eight": 8,
  "nine": 9,
}

teens = {
  "ten": 10,
  "eleven": 11,
  "twelve": 12,
  "thirteen": 13,
  "fourteen": 14,
  "fifteen": 15,
  "sixteen": 16,
  "seventeen": 17,
  "eighteen": 18,
  "nineteen": 19,
}

tens = {
  "twenty": 20,
  "thirty": 30,
  "forty": 40,
  "fifty": 50,
  "sixty": 60,
  "seventy": 70,
  "eighty": 80,
  "ninety": 90,
}
```

The screenshot shows the 'Output' tab of a Codewars kata page for '4 kyu parseint()'. It displays test results with two failures. The first failure is an 'Unexpected exception raised' in the 'Fixed tests' section, specifically in the 'Test' subsection. The traceback shows a 'NameError: name 'parse\_int' is not defined' at line 7 of 'it\_1'. The second failure is also an 'Unexpected exception raised' in the 'Random tests' section, with a similar 'NameError' at line 73 of 'it\_1'. The page also shows statistics: 2188 stars, 362 likes, 93% of 1,776 solves, 6,091 of 16,248 attempts, and 5 issues reported.

```
4 kyu parseint() reloaded
☆ 2188 ● 362 ↗ 93% of 1,776 ● 6,091 of 16,248 👤 BattleRattle ⚠ 5 Issues Reported

Instructions Output

Time: 507ms Passed: 0 Failed: 4 Errors: 2 Exit Code: 1

Test Results:
  Fixed tests
    Tests
      Unexpected exception raised
        Traceback (most recent call last):
          File "/workspace/default/.venv/lib/python3.11/site-packages/codewars_test_func()", line 7, in it_1
            test.assert_equals(parse_int('zero'), 0)
            ^^^^^^^^^^^^^
        NameError: name 'parse_int' is not defined
        Completed in 2.34ms
      Completed in 2.37ms
    Random tests
      Tests
        Unexpected exception raised
          Traceback (most recent call last):
            File "/workspace/default/.venv/lib/python3.11/site-packages/codewars_test_func()", line 73, in it_1
              test.assert_equals(parse_int(line), solution(line))
              ^^^^^^^^^^^^^
          NameError: name 'parse_int' is not defined
          Completed in 0.38ms
```

	<pre>hundreds = {     "hundred": 100, }  thousands = {     "thousand": 1000, }  millions = {     "million": 1000000, }  def word_to_num(words):     """     This function converts a     string in words to an integer.      Args:         words: A string         representing a number in         words.      Returns:         An integer equivalent to         the number in words.     """     words = words.strip().lower().split() # Preprocess the string     result = 0     i = 0     while i &lt; len(words):         if words[i] in millions:             result += millions[words[i]]         * int_from_group(words[i + 1:])</pre>	
--	--	--



	<pre>i += 2 elif words[i] in thousands:     result += thousands[words[i]] * int_from_group(words[i + 1:]) i += 2 elif words[i] in hundreds:     result += hundreds[words[i]] * int_from_group(words[i + 1:]) i += 2 elif i + 1 &lt; len(words) and words[i + 1] in tens: # Handle tens (twenty, thirty, etc.)     result += tens[words[i + 1]]     if words[i] != "zero": # Handle "twenty one" vs. "one"         result += int_from_group(words[i:]) i += 2 elif i + 1 &lt; len(words) and words[i + 1] in teens: # Handle teens (eleven, twelve, etc.)     result += teens[words[i + 1]] i += 2 elif words[i] in ones:     result += ones[words[i]] i += 1 else:     raise ValueError(f"Invalid word: {words[i]}") # Handle unexpected word  return result</pre>	
--	--	--

	<pre>def int_from_group(group):     """     This function processes a     group of words (e.g., "one     hundred") and returns the     corresponding integer.     """     if len(group) == 0:         return 0     if len(group) == 1:         return ones[group[0]]     # Handle optional "and"     between words     if group[1] == "and":         return ones[group[0]] +         int_from_group(group[2:])     else:         return         int_from_group(group[1:])</pre>	
--	---	--

Your task in order to complete this Kata is to write a function which formats a duration, given as a number of seconds, in a human-friendly way.

The function must accept a non-negative integer. If it is zero, it just returns "now". Otherwise, the duration is expressed as a combination

of years, days, hours, minutes and seconds.

It is much easier to understand with an example:

\* For seconds = 62, your function should return

"1 minute and 2 seconds"

\* For seconds = 3662, your function should return

"1 hour, 1 minute and 2 seconds"

For the purpose of this Kata, a year is 365 days and a day is 24 hours.

Note that spaces are important.

Detailed rules

The resulting expression is made of components like 4 seconds, 1 year, etc. In general, a positive integer and one of the valid units

```
def
format_duration(seconds):
    """
    This function formats a
    duration (given as seconds) in
    a human-friendly way.
```

Args:

seconds: A non-negative integer representing the duration in seconds.

Returns:

A string containing the human-readable duration.

```
"""
units = [("year", 365 * 24 *
60 * 60), ("day", 24 * 60 * 60),
("hour", 60 * 60), ("minute",
60), ("second", 1)]
results = []
for unit, value in units:
    if seconds >= value:
        count = seconds // value
        seconds -= count * value
        results.append(f'{count}
{unit + ("s" if count > 1 else
"")}')
    return "now" if not results
else " , ".join(results[:-1]) + "
and " + results[-1]
```

4 kyu Human readable duration format

☆ 3755 🗳 762 🔄 92% of 7,431 🕒 27,297 of 76,918 👤 davazp 🚩 4 Issues Reported

Instructions Output Past Solutions

Time: 508ms Passed: 110 Failed: 3 Exit Code: 1

Test Results:

Fixed Tests

Basic Test Cases

- Test Passed
- ! ' and 1 second' should equal '1 second'
- Test Passed
- ! ' and 2 minutes' should equal '2 minutes'
- ! ' and 1 hour' should equal '1 hour'
- Test Passed
- Test Passed
- Test Passed
- Test Passed
- Test Passed
- Test Passed
- Test Passed

Completed in 0.18ms

Completed in 0.21ms

Random Tests

<p>of time, separated by a space. The unit of time is used in plural if the integer is greater than 1. The components are separated by a comma and a space (" "). Except the last component, which is separated by " and ", just like it would be written in English. A more significant units of time will occur before than a least significant one. Therefore, 1 second and 1 year is not correct, but 1 year and 1 second is. Different components have different unit of times. So there is not repeated units like in 5 seconds and 1 second. A component will not appear at all if its value happens to be zero. Hence, 1 minute and 0 seconds is not valid, but it should be just 1 minute. A unit of time must be used "as much as possible". It means that the function should not return 61 seconds, but 1 minute and 1 second instead. Formally, the duration specified by of a component must not be greater</p>		
---	--	--

than any valid more significant unit of time.		
--	--	--