

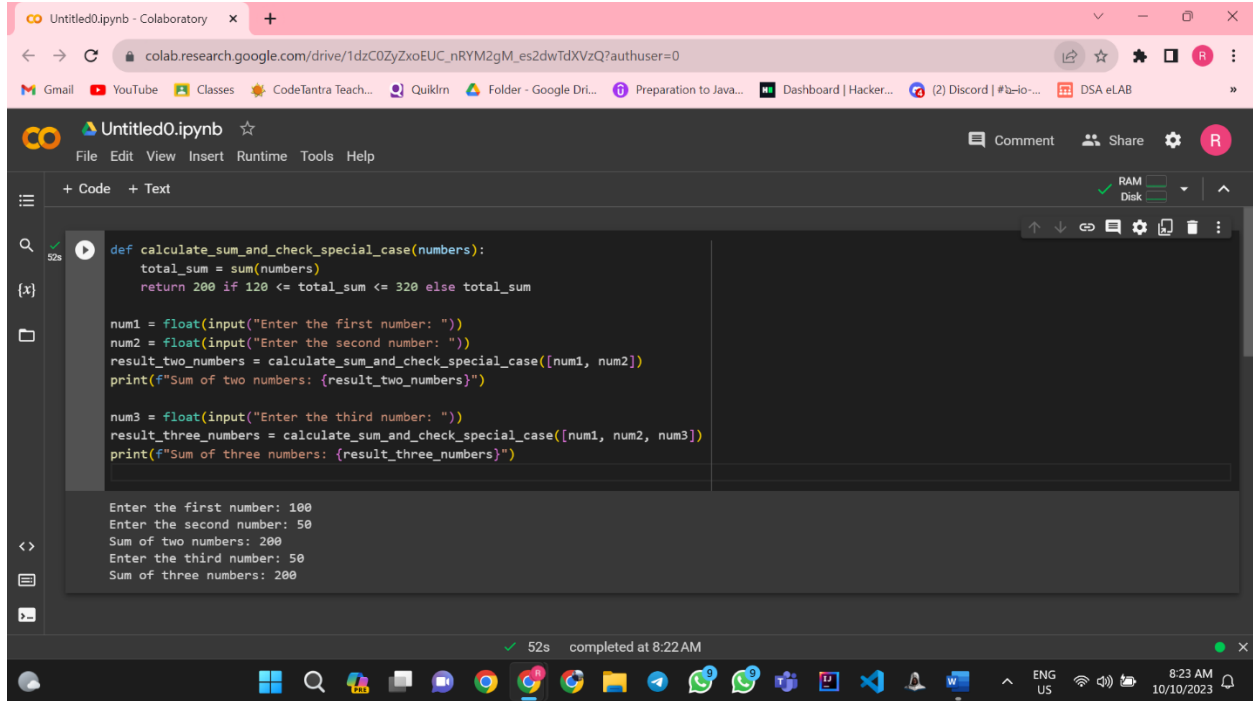
Rohan Soni

RA2211003012027

Week 10

1. Write a python program to calculate the sum of Two numbers and Three numbers.

However, if the sum is between 120 to 320 it will return 200.



The screenshot shows a Google Colaboratory notebook titled 'Untitled0.ipynb'. The code is as follows:

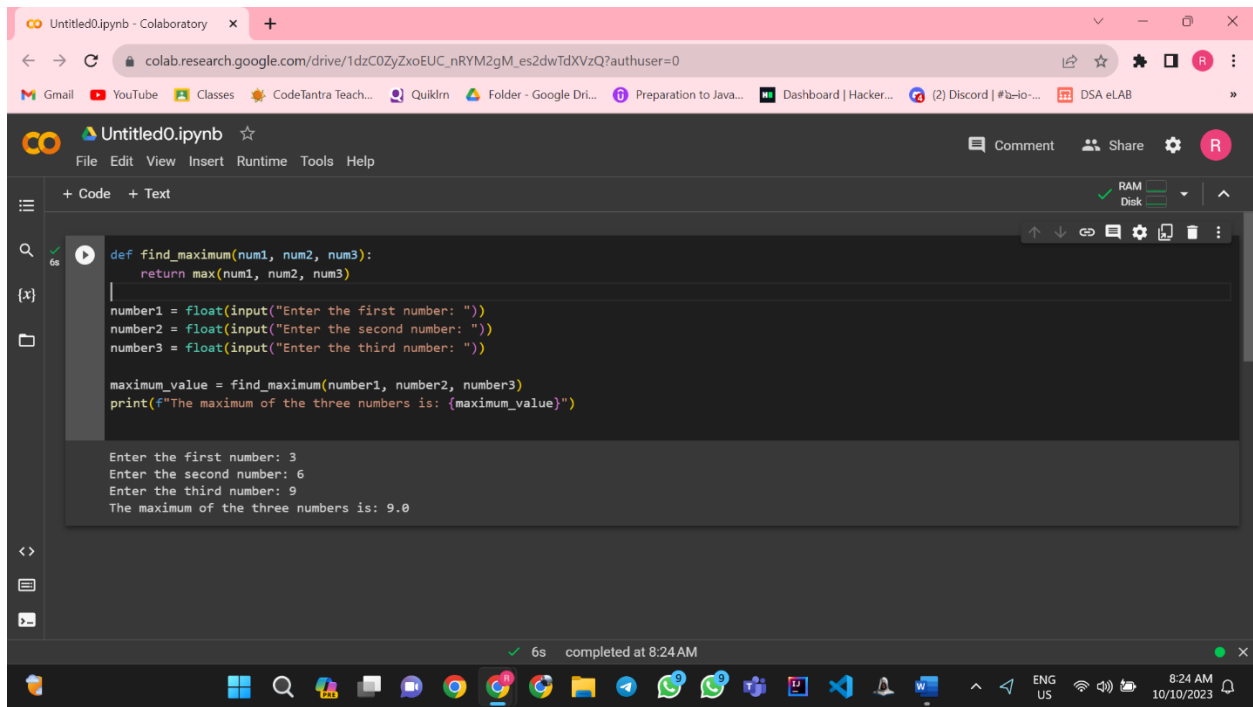
```
def calculate_sum_and_check_special_case(numbers):  
    total_sum = sum(numbers)  
    return 200 if 120 <= total_sum <= 320 else total_sum  
  
num1 = float(input("Enter the first number: "))  
num2 = float(input("Enter the second number: "))  
result_two_numbers = calculate_sum_and_check_special_case([num1, num2])  
print(f"Sum of two numbers: {result_two_numbers}")  
  
num3 = float(input("Enter the third number: "))  
result_three_numbers = calculate_sum_and_check_special_case([num1, num2, num3])  
print(f"Sum of three numbers: {result_three_numbers}")
```

The output of the program is shown below the code:

```
Enter the first number: 100  
Enter the second number: 50  
Sum of two numbers: 200  
Enter the third number: 50  
Sum of three numbers: 200
```

The notebook interface includes a menu bar (File, Edit, View, Insert, Runtime, Tools, Help), a toolbar with icons for file operations, and a status bar at the bottom indicating the execution time (52s) and completion status (completed at 8:22 AM).

2. Implement a python function to find the Maximum of Three numbers.



The screenshot shows a Google Colaboratory notebook interface. The browser address bar displays the URL: `colab.research.google.com/drive/1dzC0ZyZxoEUC_nRYM2gM_es2dwTdXVzQ?authuser=0`. The notebook is titled "Untitled0.ipynb". The code editor contains the following Python code:

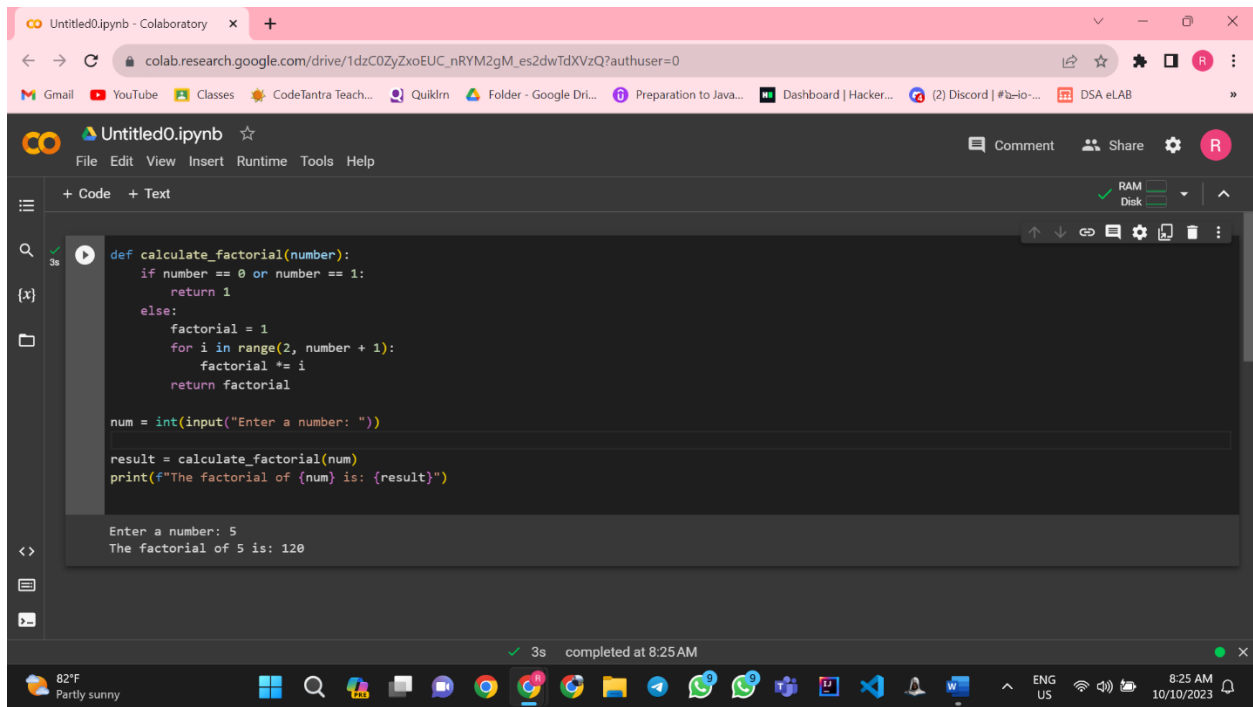
```
def find_maximum(num1, num2, num3):  
    return max(num1, num2, num3)  
  
number1 = float(input("Enter the first number: "))  
number2 = float(input("Enter the second number: "))  
number3 = float(input("Enter the third number: "))  
  
maximum_value = find_maximum(number1, number2, number3)  
print(f"The maximum of the three numbers is: {maximum_value}")
```

The output of the code execution is displayed below the code cell:

```
Enter the first number: 3  
Enter the second number: 6  
Enter the third number: 9  
The maximum of the three numbers is: 9.0
```

The status bar at the bottom indicates that the code was executed successfully, with a green checkmark and the text "6s completed at 8:24 AM". The system tray at the bottom shows various icons, including the Windows Start button, search, task view, and several application icons, along with the date and time "8:24 AM 10/10/2023".

3. Write a python program to calculate the Factorial of a given number.



The screenshot shows a Google Colaboratory notebook titled 'Untitled0.ipynb'. The notebook contains a Python program to calculate the factorial of a given number. The code is as follows:

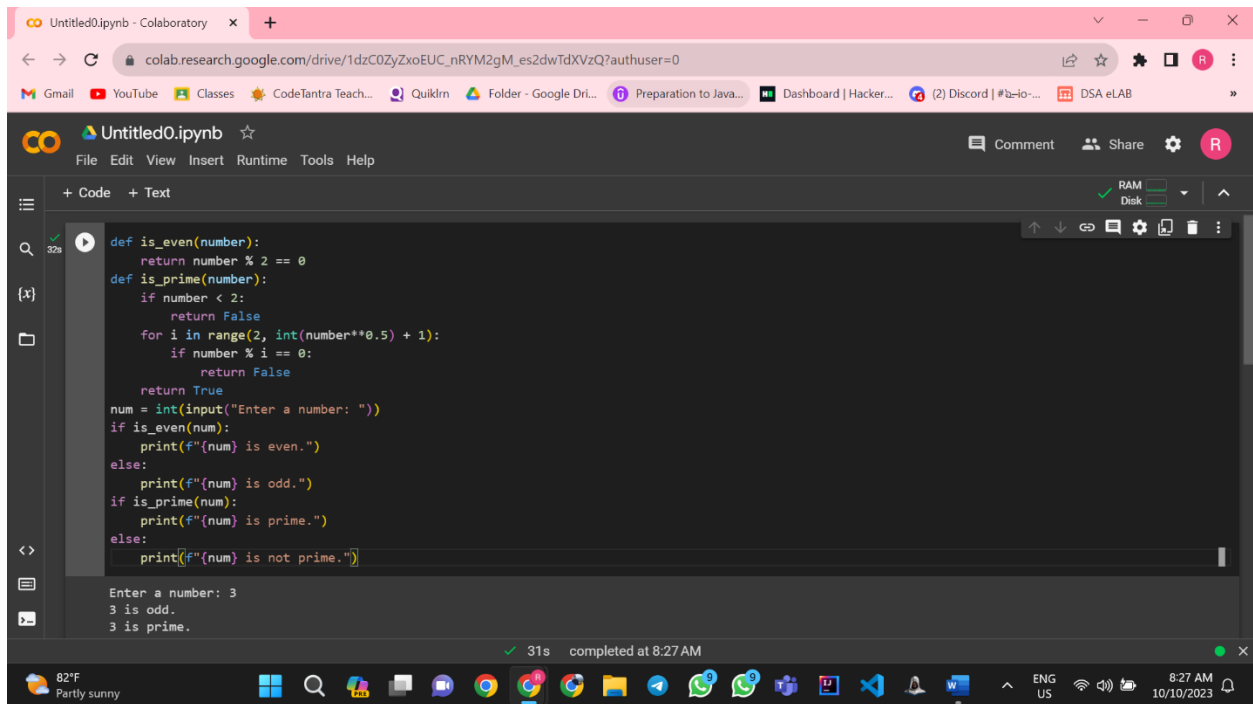
```
def calculate_factorial(number):  
    if number == 0 or number == 1:  
        return 1  
    else:  
        factorial = 1  
        for i in range(2, number + 1):  
            factorial *= i  
        return factorial  
  
num = int(input("Enter a number: "))  
  
result = calculate_factorial(num)  
print(f"The factorial of {num} is: {result}")
```

The output of the program is shown in the output cell:

```
Enter a number: 5  
The factorial of 5 is: 120
```

The notebook interface includes a menu bar with options like File, Edit, View, Insert, Runtime, Tools, and Help. The bottom status bar shows the execution time as 3s and the completion time as 8:25 AM.

4. Write a python program to Check if a Number is Even or Odd and also check whether it is Prime or not.



The screenshot shows a Google Colaboratory notebook titled "Untitled0.ipynb". The code in the notebook is as follows:

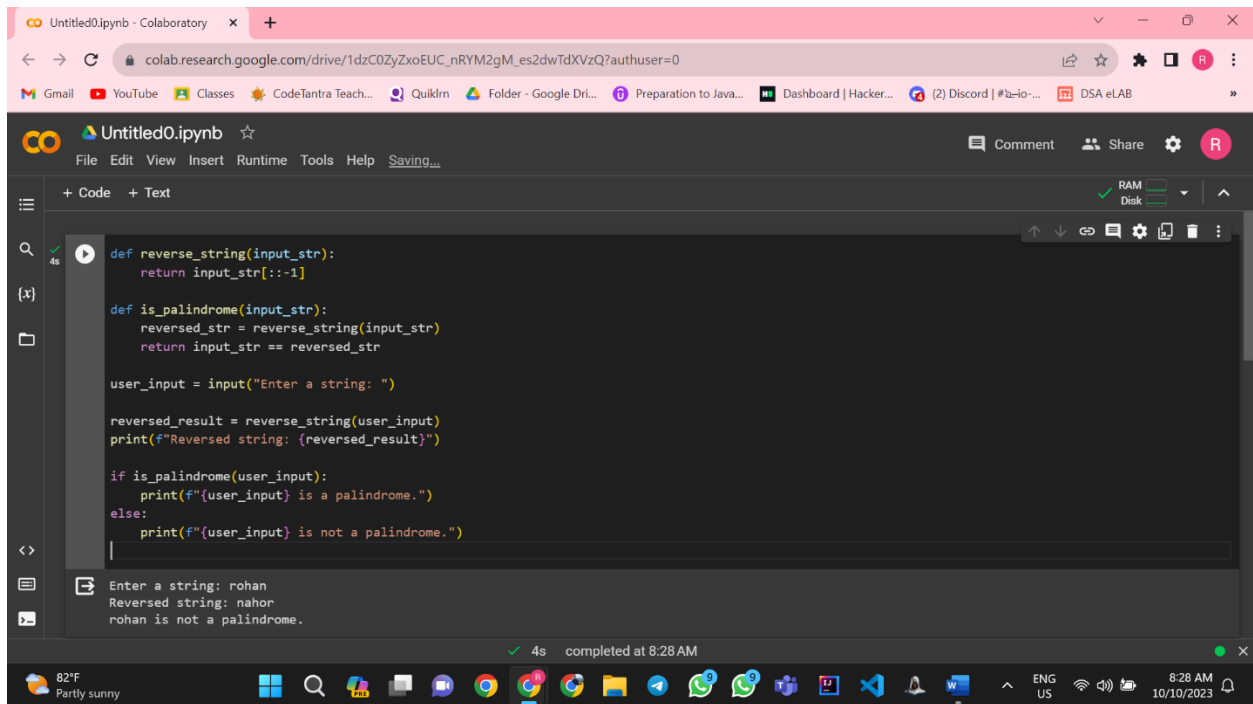
```
def is_even(number):  
    return number % 2 == 0  
  
def is_prime(number):  
    if number < 2:  
        return False  
    for i in range(2, int(number**0.5) + 1):  
        if number % i == 0:  
            return False  
    return True  
  
num = int(input("Enter a number: "))  
if is_even(num):  
    print(f"{num} is even.")  
else:  
    print(f"{num} is odd.")  
if is_prime(num):  
    print(f"{num} is prime.")  
else:  
    print(f"{num} is not prime.")
```

The output of the program is:

```
Enter a number: 3  
3 is odd.  
3 is prime.
```

The notebook interface includes a menu bar (File, Edit, View, Insert, Runtime, Tools, Help), a toolbar with icons for running, saving, and sharing, and a status bar at the bottom showing the execution time (31s) and completion status (completed at 8:27 AM).

5. Implement a python function to Reverse a given String and also check for palindrome or not.



The screenshot shows a Google Colaboratory notebook titled 'Untitled0.ipynb'. The code in the notebook is as follows:

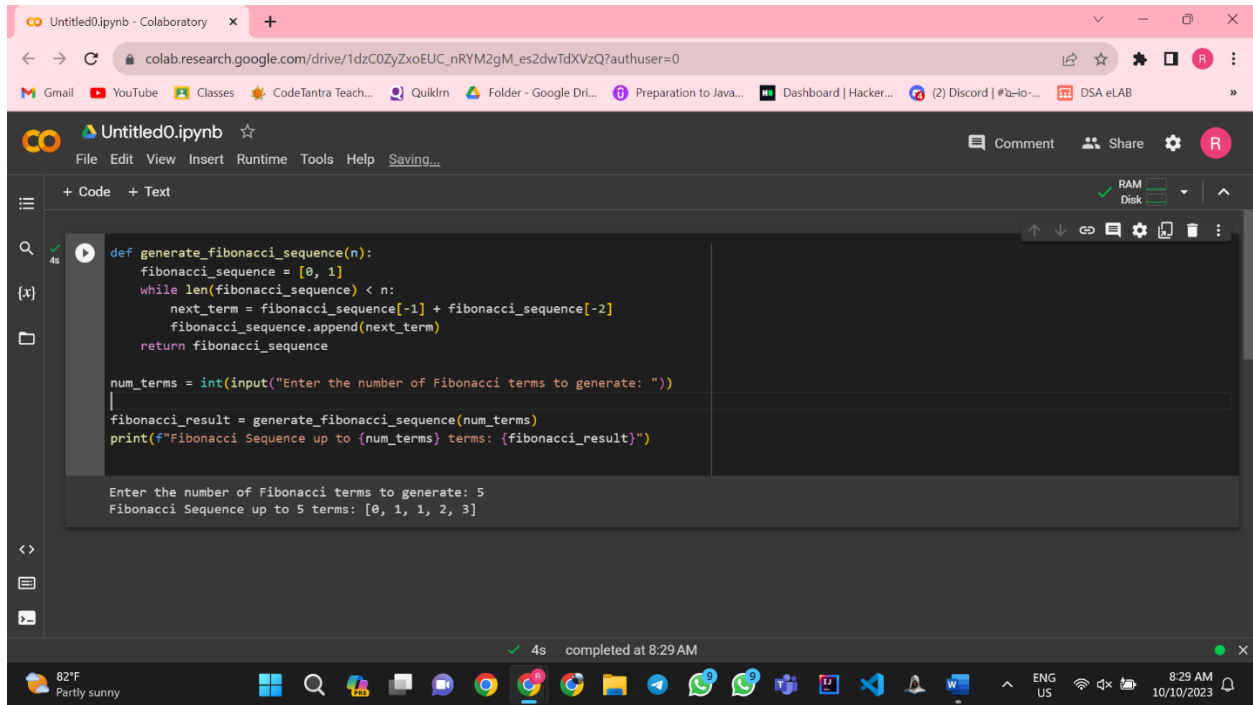
```
def reverse_string(input_str):  
    return input_str[::-1]  
  
def is_palindrome(input_str):  
    reversed_str = reverse_string(input_str)  
    return input_str == reversed_str  
  
user_input = input("Enter a string: ")  
  
reversed_result = reverse_string(user_input)  
print(f"Reversed string: {reversed_result}")  
  
if is_palindrome(user_input):  
    print(f"{user_input} is a palindrome.")  
else:  
    print(f"{user_input} is not a palindrome.")
```

The output of the code is displayed below the code cell:

```
Enter a string: rohan  
Reversed string: nahor  
rohan is not a palindrome.
```

The notebook interface includes a menu bar (File, Edit, View, Insert, Runtime, Tools, Help), a toolbar with icons for search, zoom, and other functions, and a status bar at the bottom showing the temperature (82°F), weather (Partly sunny), and the time (8:28 AM 10/10/2023).

6. Write a python program to Generate Fibonacci Sequence.



The screenshot shows a Google Colaboratory notebook titled "Untitled0.ipynb". The notebook contains a Python program that generates a Fibonacci sequence. The code is as follows:

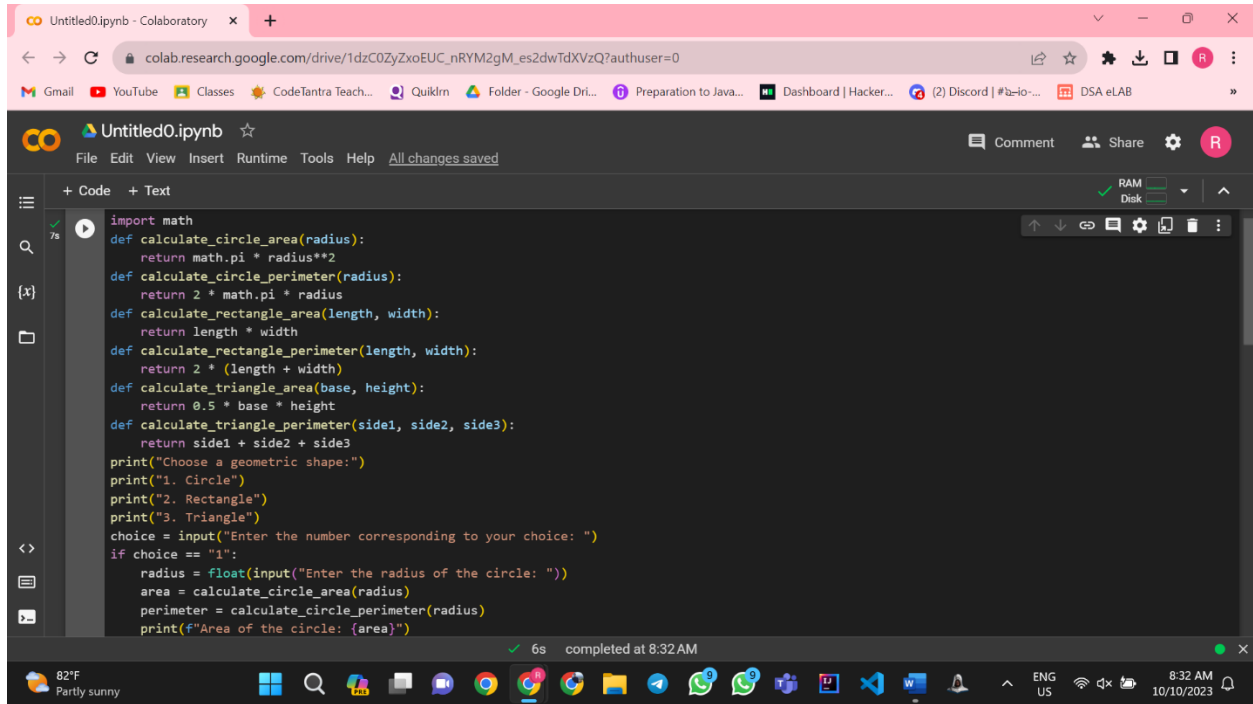
```
def generate_fibonacci_sequence(n):  
    fibonacci_sequence = [0, 1]  
    while len(fibonacci_sequence) < n:  
        next_term = fibonacci_sequence[-1] + fibonacci_sequence[-2]  
        fibonacci_sequence.append(next_term)  
    return fibonacci_sequence  
  
num_terms = int(input("Enter the number of Fibonacci terms to generate: "))  
fibonacci_result = generate_fibonacci_sequence(num_terms)  
print(f"Fibonacci Sequence up to {num_terms} terms: {fibonacci_result}")
```

The output of the program is displayed below the code:

```
Enter the number of Fibonacci terms to generate: 5  
Fibonacci Sequence up to 5 terms: [0, 1, 1, 2, 3]
```

The notebook interface includes a menu bar with options like File, Edit, View, Insert, Runtime, Tools, and Help. The status bar at the bottom indicates that the code was completed at 8:29 AM on 10/10/2023.

7. Write a python program to calculate the area and perimeter of different geometric shapes (circle, rectangle, triangle, etc.).



The screenshot shows a Google Colaboratory notebook titled 'Untitled0.ipynb'. The code defines functions for calculating the area and perimeter of circles, rectangles, and triangles. It prompts the user to choose a shape and then calculates the area for a circle based on the chosen option.

```
import math
def calculate_circle_area(radius):
    return math.pi * radius**2
def calculate_circle_perimeter(radius):
    return 2 * math.pi * radius
def calculate_rectangle_area(length, width):
    return length * width
def calculate_rectangle_perimeter(length, width):
    return 2 * (length + width)
def calculate_triangle_area(base, height):
    return 0.5 * base * height
def calculate_triangle_perimeter(side1, side2, side3):
    return side1 + side2 + side3
print("Choose a geometric shape:")
print("1. Circle")
print("2. Rectangle")
print("3. Triangle")
choice = input("Enter the number corresponding to your choice: ")
if choice == "1":
    radius = float(input("Enter the radius of the circle: "))
    area = calculate_circle_area(radius)
    perimeter = calculate_circle_perimeter(radius)
    print(f"Area of the circle: {area}")
```

The notebook interface includes a file explorer on the left, a toolbar at the top with options like 'File', 'Edit', 'View', 'Insert', 'Runtime', 'Tools', and 'Help', and a status bar at the bottom showing '6s completed at 8:32 AM'.

Untitled0.ipynb - Colaboratory

colab.research.google.com/drive/1dzC0ZyZxoEUC_nRYM2gM_es2dwTdXVzQ?authuser=0

82°F Partly sunny

6s completed at 8:32 AM

```
print(f"Perimeter of the circle: {perimeter}")
elif choice == "2":
    length = float(input("Enter the length of the rectangle: "))
    width = float(input("Enter the width of the rectangle: "))
    area = calculate_rectangle_area(length, width)
    perimeter = calculate_rectangle_perimeter(length, width)
    print(f"Area of the rectangle: {area}")
    print(f"Perimeter of the rectangle: {perimeter}")
elif choice == "3":
    base = float(input("Enter the base of the triangle: "))
    height = float(input("Enter the height of the triangle: "))
    area = calculate_triangle_area(base, height)
    print(f"Area of the triangle: {area}")
    side1 = float(input("Enter the length of side 1: "))
    side2 = float(input("Enter the length of side 2: "))
    side3 = float(input("Enter the length of side 3: "))
    perimeter = calculate_triangle_perimeter(side1, side2, side3)
    print(f"Perimeter of the triangle: {perimeter}")
else:
    print("Invalid choice. Please choose a number from 1 to 3.")
```

Choose a geometric shape:
1. Circle

Untitled0.ipynb - Colaboratory

colab.research.google.com/drive/1dzC0ZyZxoEUC_nRYM2gM_es2dwTdXVzQ?authuser=0

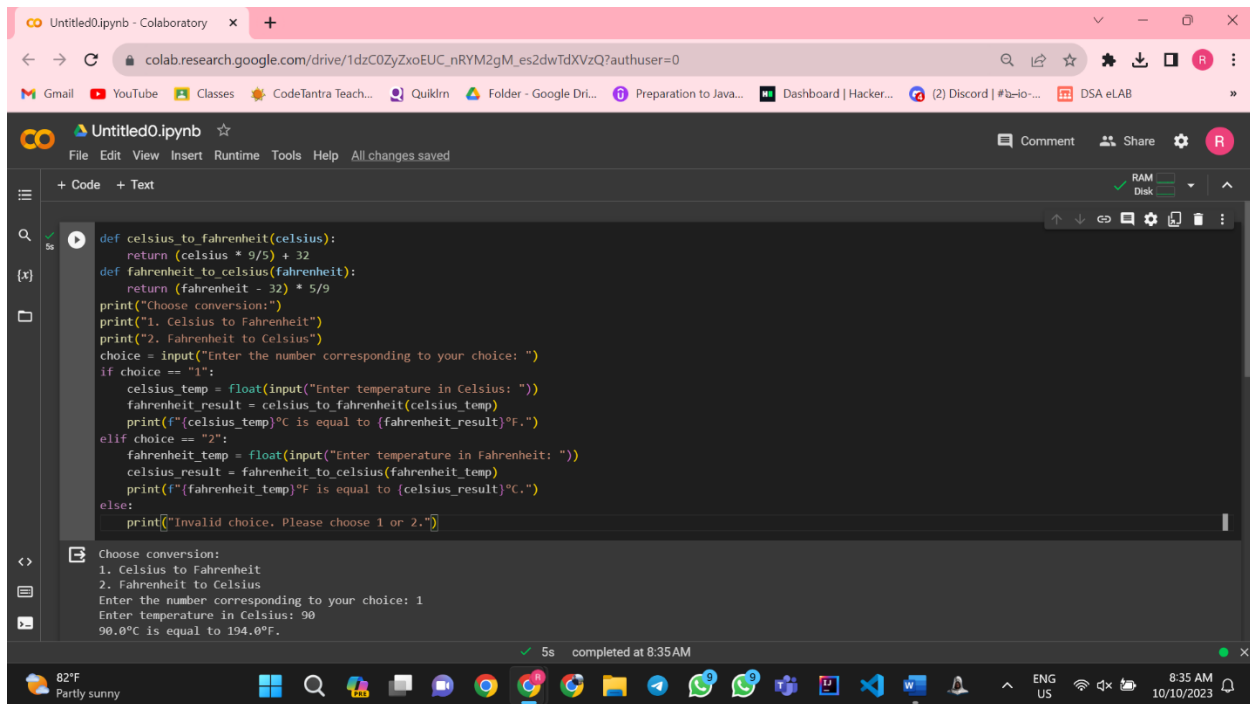
82°F Partly sunny

6s completed at 8:32 AM

```
print(f"Area of the rectangle: {area}")
print(f"Perimeter of the rectangle: {perimeter}")
elif choice == "3":
    base = float(input("Enter the base of the triangle: "))
    height = float(input("Enter the height of the triangle: "))
    area = calculate_triangle_area(base, height)
    print(f"Area of the triangle: {area}")
    side1 = float(input("Enter the length of side 1: "))
    side2 = float(input("Enter the length of side 2: "))
    side3 = float(input("Enter the length of side 3: "))
    perimeter = calculate_triangle_perimeter(side1, side2, side3)
    print(f"Perimeter of the triangle: {perimeter}")
else:
    print("Invalid choice. Please choose a number from 1 to 3.")
```

Choose a geometric shape:
1. Circle
2. Rectangle
3. Triangle
Enter the number corresponding to your choice: 1
Enter the radius of the circle: 7
Area of the circle: 153.93804002589985
Perimeter of the circle: 43.982297150257104

8. Implement a python function to Convert Celsius to Fahrenheit and Fahrenheit to Celsius.

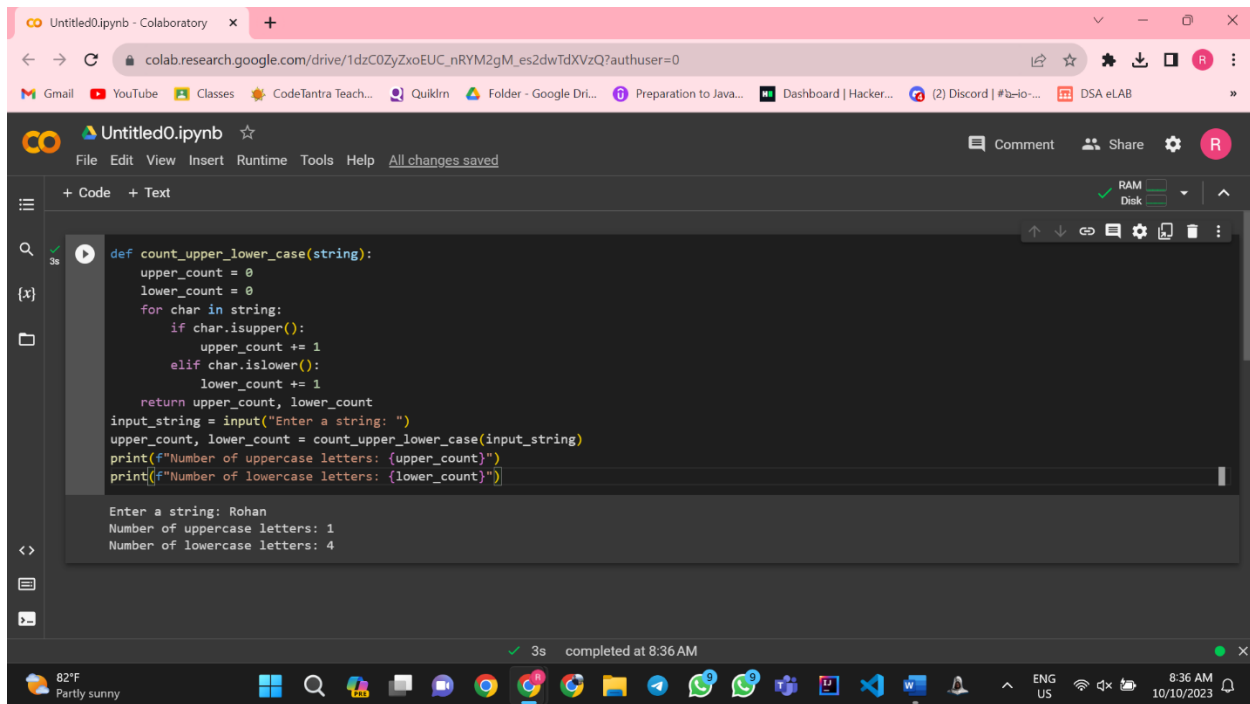


The screenshot shows a Google Colaboratory notebook titled 'Untitled0.ipynb'. The code defines two functions: `celsius_to_fahrenheit` and `fahrenheit_to_celsius`. It then prompts the user to choose a conversion direction and enter a temperature value. The output shows the program running successfully, displaying the conversion results for 90°C to 194.0°F.

```
def celsius_to_fahrenheit(celsius):  
    return (celsius * 9/5) + 32  
  
def fahrenheit_to_celsius(fahrenheit):  
    return (fahrenheit - 32) * 5/9  
  
print("Choose conversion:")  
print("1. Celsius to Fahrenheit")  
print("2. Fahrenheit to Celsius")  
choice = input("Enter the number corresponding to your choice: ")  
  
if choice == "1":  
    celsius_temp = float(input("Enter temperature in Celsius: "))  
    fahrenheit_result = celsius_to_fahrenheit(celsius_temp)  
    print(f"{celsius_temp}°C is equal to {fahrenheit_result}°F.")  
  
elif choice == "2":  
    fahrenheit_temp = float(input("Enter temperature in Fahrenheit: "))  
    celsius_result = fahrenheit_to_celsius(fahrenheit_temp)  
    print(f"{fahrenheit_temp}°F is equal to {celsius_result}°C.")  
  
else:  
    print("Invalid choice. Please choose 1 or 2.")
```

Choose conversion:
1. Celsius to Fahrenheit
2. Fahrenheit to Celsius
Enter the number corresponding to your choice: 1
Enter temperature in Celsius: 90
90.0°C is equal to 194.0°F.

9. Write a Python program that accepts a string and counts the number of upper and lower case letters.



The screenshot shows a Google Colaboratory notebook titled "Untitled0.ipynb". The notebook contains a Python function `count_upper_lower_case` that takes a string as input and returns the count of uppercase and lowercase letters. The function is defined as follows:

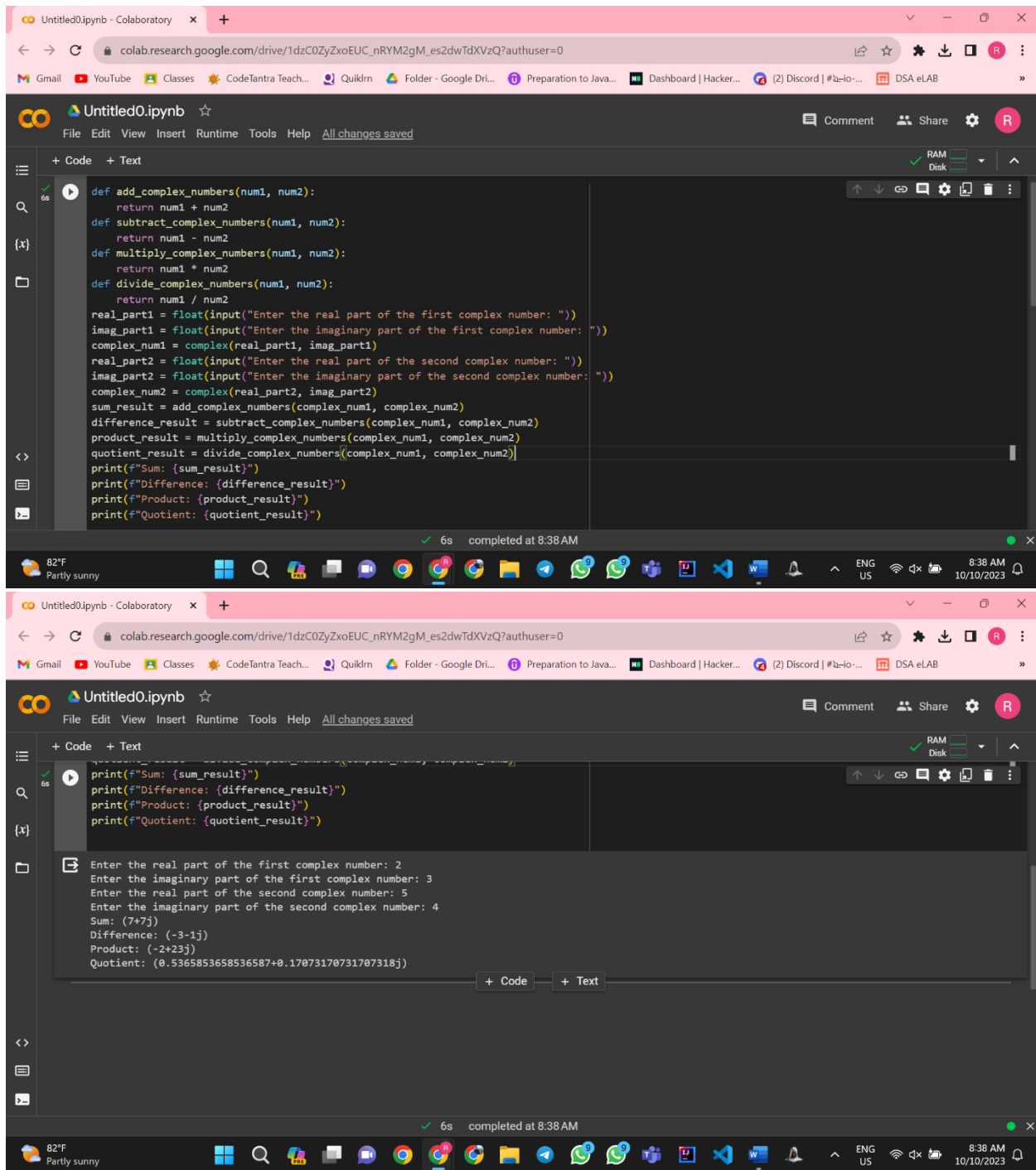
```
def count_upper_lower_case(string):  
    upper_count = 0  
    lower_count = 0  
    for char in string:  
        if char.isupper():  
            upper_count += 1  
        elif char.islower():  
            lower_count += 1  
    return upper_count, lower_count  
  
input_string = input("Enter a string: ")  
upper_count, lower_count = count_upper_lower_case(input_string)  
print(f"Number of uppercase letters: {upper_count}")  
print(f"Number of lowercase letters: {lower_count}")
```

The notebook shows the output of the program when the input string is "Rohan":

```
Enter a string: Rohan  
Number of uppercase letters: 1  
Number of lowercase letters: 4
```

The notebook interface includes a menu bar with options like File, Edit, View, Insert, Runtime, Tools, and Help. The status bar at the bottom indicates that the code was completed at 8:36 AM on 10/10/2023.

10. Write a python program to perform Arithmetic operations on Complex Numbers.



The image displays two screenshots of a Google Colab notebook titled 'Untitled0.ipynb'. The top screenshot shows the code for performing arithmetic operations on complex numbers. The code defines four functions: `add_complex_numbers`, `subtract_complex_numbers`, `multiply_complex_numbers`, and `divide_complex_numbers`. It then prompts the user to enter the real and imaginary parts of two complex numbers, constructs them as `complex` objects, and performs the four operations. The results are printed in a formatted manner.

```
def add_complex_numbers(num1, num2):  
    return num1 + num2  
def subtract_complex_numbers(num1, num2):  
    return num1 - num2  
def multiply_complex_numbers(num1, num2):  
    return num1 * num2  
def divide_complex_numbers(num1, num2):  
    return num1 / num2  
  
real_part1 = float(input("Enter the real part of the first complex number: "))  
imag_part1 = float(input("Enter the imaginary part of the first complex number: "))  
complex_num1 = complex(real_part1, imag_part1)  
real_part2 = float(input("Enter the real part of the second complex number: "))  
imag_part2 = float(input("Enter the imaginary part of the second complex number: "))  
complex_num2 = complex(real_part2, imag_part2)  
sum_result = add_complex_numbers(complex_num1, complex_num2)  
difference_result = subtract_complex_numbers(complex_num1, complex_num2)  
product_result = multiply_complex_numbers(complex_num1, complex_num2)  
quotient_result = divide_complex_numbers(complex_num1, complex_num2)  
print(f"Sum: {sum_result}")  
print(f"Difference: {difference_result}")  
print(f"Product: {product_result}")  
print(f"Quotient: {quotient_result}")
```

The bottom screenshot shows the output of the program. The user has entered the real and imaginary parts for two complex numbers: (2, 3) and (5, 4). The program outputs the results of the four operations.

```
Enter the real part of the first complex number: 2  
Enter the imaginary part of the first complex number: 3  
Enter the real part of the second complex number: 5  
Enter the imaginary part of the second complex number: 4  
Sum: (7+7j)  
Difference: (-3-1j)  
Product: (-2+23j)  
Quotient: (0.5365853658536587+0.17073170731707318j)
```

HACKERRANK

<https://www.hackerrank.com/challenges/default-arguments/problem?isFullScreen=true>

HackerRank Prepare > Python > Debugging > Default Arguments Exit Full Screen View

Problem

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

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

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

The function works like this:

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

Submissions

```
class EvenStream(object):
    def __init__(self):
        self.current = 0
    def get_next(self):
        to_return = self.current
        self.current += 2
        return to_return
class OddStream(object):
    def __init__(self):
        self.current = 1
    def get_next(self):
        to_return = self.current
        self.current += 2
        return to_return
def print_from_stream(n, stream=None):
    if stream is None:
        stream = EvenStream()
```

Line: 28 Col: 42

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

Leaderboard

```
if stream is None:
    stream = EvenStream()
for _ in range(n):
    print(stream.get_next())
raw_input = input
queries = int(input())
for _ in range(queries):
    stream_name, n = input().split()
    n = int(n)
    if stream_name == "even":
        print_from_stream(n)
    else:
        print_from_stream(n, OddStream())
```

Line: 28 Col: 42

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

You have earned 30.00 points!
9/115 challenges solved. 8%

82°F Partly sunny 8:45 AM 10/10/2023

<https://www.hackerrank.com/challenges/words-score/problem?isFullScreen=true>

The screenshot displays the HackerRank 'Words Score' challenge in full-screen mode. The interface is divided into three main sections: Problem, Submissions, and Leaderboard. The Problem section contains the following text:

successfully execute all provided test files.

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

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

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

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

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

Input Format

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

The code editor shows the following Python code:

```
def is_vowel(letter):
    return letter in ['a', 'e', 'i', 'o', 'u', 'y']
def score_words(words):
    score = 0
    for word in words:
        num_vowels = 0
        for letter in word:
            if is_vowel(letter):
                num_vowels += 1
        if num_vowels % 2 == 0:
            score += 2
        else:
            score += 1
    return score
n = int(input())
words = input().split()
print(score_words(words))
```

The bottom section of the interface displays a success message: "You have earned 10.00 points! 10/115 challenges solved. 99%". Below this, a green banner reads "Congratulations You solved this challenge. Would you like to challenge your friends?" with social media icons for Facebook, Twitter, and LinkedIn. A "Next Challenge" button is also present.

<https://www.hackerrank.com/challenges/reduce-function/problem?isFullScreen=true>

The image shows a screenshot of the HackerRank website interface for the 'Reduce Function' challenge. The interface is divided into several sections:

- Problem:** Contains sample input and output. The input is a lambda function `lambda x, y : x + y` applied to the list `[1,2,3]` with an initial value of `-3`, resulting in `3`. The second input is `from fractions import gcd` followed by `reduce(gcd, [2,4,8], 3)`, resulting in `1`.
- Input Format:** States that the first line contains n , the number of rational numbers. The next n lines contain two integers each, the numerator (N_i) and denominator (D_i) of the i^{th} rational number.
- Constraints:**
 - $1 \leq n \leq 100$
 - $1 \leq N_i, D_i \leq 10^9$
- Output Format:** Requires printing only one line containing the numerator and denominator of the product of the numbers in the list in its simplest form.

The right side of the interface shows a code editor with the following Python code:

```
from fractions import Fraction
from functools import reduce

def product(fracs):
    t = Fraction(reduce(lambda x, y: x * y, fracs))
    return t.numerator, t.denominator

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

Below the code editor, there are buttons for 'Upload Code as File', 'Test against custom input', 'Run Code', and 'Submit Code'. The 'Submit Code' button is highlighted in green.

At the bottom, a confirmation message states: 'You have earned 30.00 points! 11/115 challenges solved. 10%'. Below this is a green banner with 'Congratulations' and 'You solved this challenge. Would you like to challenge your friends?'. There are social media icons for Facebook, Twitter, and LinkedIn, and a 'Next Challenge' button.

On the left side of the confirmation message, there is a list of test cases: 'Test case 0', 'Test case 1', 'Test case 2', and 'Test case 3', each with a green checkmark and a lock icon.

Below the test cases, there is a 'Compiler Message' section showing 'Success'. At the bottom, there is a 'Hidden Test Case' section with the text 'Unlock this test case for 5 hacks.'.