***Python Assignment Questions***

1. **File Analysis: Count Words, Lines, and Characters**

**Objective**:

Write a Python script that reads a text file and computes:

1. The total number of lines.
2. The total number of words.
3. The total number of characters.

**Requirements**:

1. Accept the file path as input.
2. Handle cases where the file does not exist gracefully.

import os

def file\_analysis(file\_path):

try:

with open(file\_path, 'r') as file:

lines = file.readlines()

num\_lines = len(lines)

num\_words = sum(len(line.split()) for line in lines)

num\_characters = sum(len(line) for line in lines)

print(f"Total lines: {num\_lines}")

print(f"Total words: {num\_words}")

print(f"Total characters: {num\_characters}")

except FileNotFoundError:

print(f"The file at {file\_path} does not exist.")

# Example usage

file\_path = input("Enter the file path: ")

file\_analysis(file\_path)

1. **Generate Fibonacci Sequence up to N and Find Primes**

**Objective**:

Write a Python function that:

1. Generates the Fibonacci sequence up to a given number N.
2. Filters and prints only the prime numbers from the sequence.

def is\_prime(num):

if num <= 1:

return False

for i in range(2, int(num\*\*0.5) + 1):

if num % i == 0:

return False

return True

def fibonacci\_primes(N):

fib\_sequence = [0, 1]

while fib\_sequence[-1] <= N:

fib\_sequence.append(fib\_sequence[-1] + fib\_sequence[-2])

prime\_fibs = [num for num in fib\_sequence if is\_prime(num) and num <= N]

print("Fibonacci numbers up to", N, ":", fib\_sequence[:-1]) # Exclude the last number that exceeds N

print("Prime Fibonacci numbers:", prime\_fibs)

# Example usage

N = int(input("Enter the value of N: "))

fibonacci\_primes(N)

**3. Find the Second Largest Number in a List**  
  
 **Objective**:

Write a Python function that takes a list of integers as input and returns the second largest number in the list.

**Requirements**:

1. Handle cases where the list has duplicates.
2. If the list has fewer than two unique numbers, return None.

def second\_largest(nums):

unique\_nums = list(set(nums)) # Remove duplicates

if len(unique\_nums) < 2:

return None

unique\_nums.sort(reverse=True) # Sort in descending order

return unique\_nums[1]

# Example usage

nums = list(map(int, input("Enter a list of integers: ").split()))

result = second\_largest(nums)

if result is None:

print("The list has fewer than two unique numbers.")

else:

print("The second largest number is:", result)

4. **Check If a String is a Valid Anagram**  
  
**Objective**: Write a Python function that checks whether two strings are anagrams of each other.

**Requirements**:

1. Ignore case sensitivity.
2. Consider only alphanumeric characters (ignore spaces and punctuation)

import re

def is\_anagram(str1, str2):

# Remove non-alphanumeric characters and convert to lowercase

str1 = ''.join(re.findall(r'\w', str1.lower()))

str2 = ''.join(re.findall(r'\w', str2.lower()))

return sorted(str1) == sorted(str2)

# Example usage

str1 = input("Enter the first string: ")

str2 = input("Enter the second string: ")

if is\_anagram(str1, str2):

print("The strings are anagrams.")

else:

print("The strings are not anagrams.")