ASSIGNMENT-1.4

2303A51337

BATCH-10

➢ TASK-1:

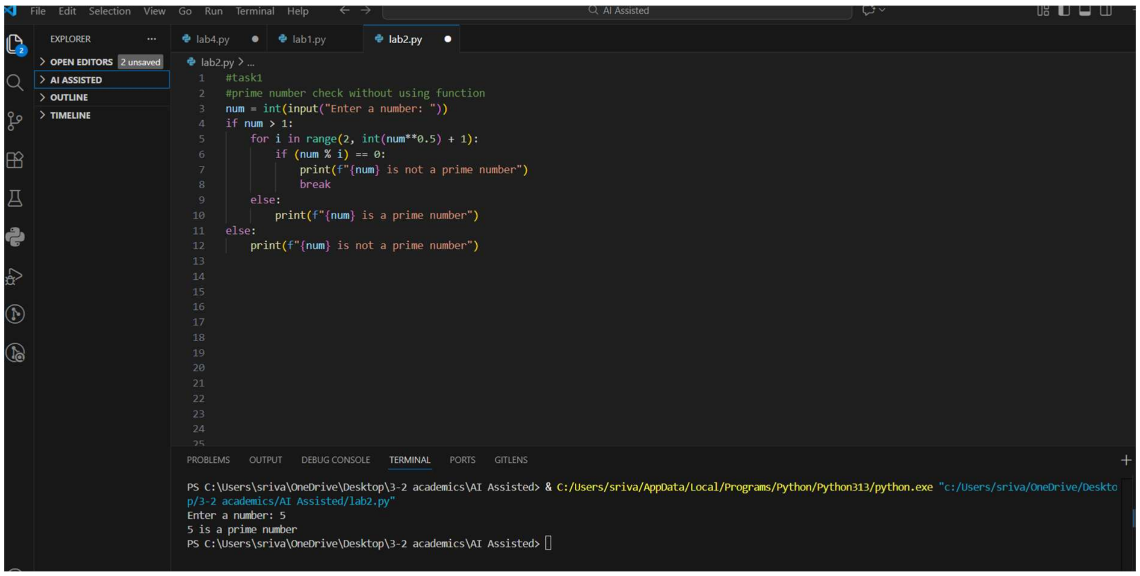
PROMPT:

Prime number check without using function.

CODE:

```
num = int(input("Enter a number: "))
if num > 1:
    for i in range(2, int(num**0.5) + 1):
        if (num % i) == 0:
            print(f"{num} is not a prime number")
            break
    else:
        print(f"{num} is a prime number")
else:
    print(f"{num} is not a prime number")
```

OUTPUT:



```
#task1
#prime number check without using function
num = int(input("Enter a number: "))
if num > 1:
    for i in range(2, int(num**0.5) + 1):
        if (num % i) == 0:
            print(f"{num} is not a prime number")
            break
    else:
        print(f"{num} is a prime number")
else:
    print(f"{num} is not a prime number")
```

```
PS C:\Users\sriva\OneDrive\Desktop\3-2 academics\AI Assisted> & C:/Users/sriva/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/sriva/OneDrive/Deskto
p/3-2 academics/AI Assisted/lab2.py"
Enter a number: 5
5 is a prime number
PS C:\Users\sriva\OneDrive\Desktop\3-2 academics\AI Assisted>
```

EXPLANATION:

- This program checks whether a number is prime by testing if it has any divisors from 2 to √n.
- If it is divisible by any number, it is not prime.
- If no divisor is found, it is prime.
- Numbers less than or equal to 1 are not prime..

❖ TASK-2

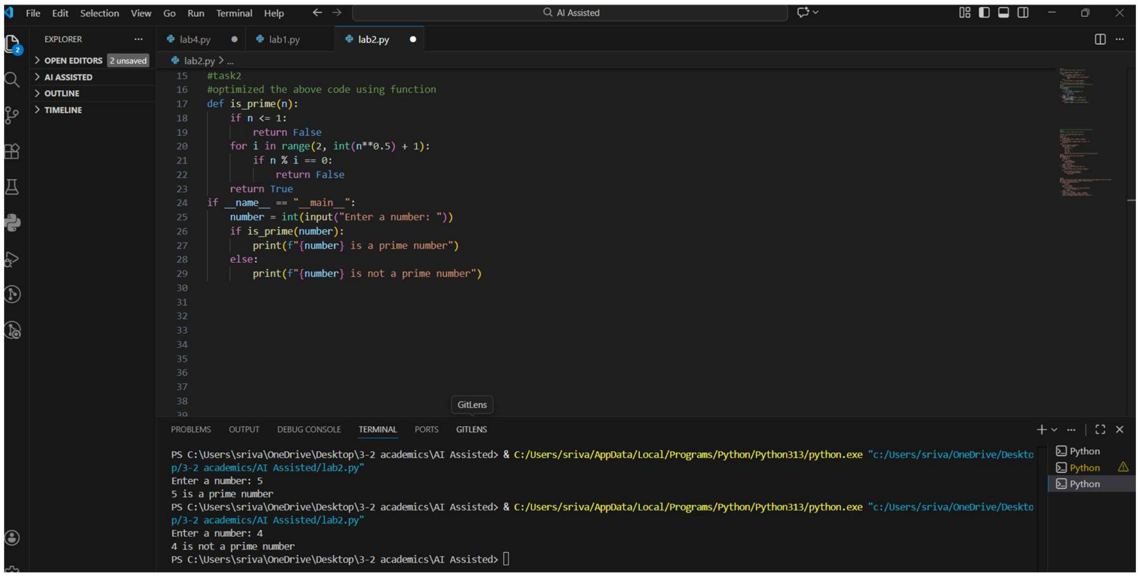
PROMPT:

optimized the above code using function.

CODE:

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

OUTPUT:



```
#task2
#optimized the above code using function
def is_prime(n):
    if n <= 1:
        return False
    for i in range(2, int(n**0.5) + 1):
        if n % i == 0:
            return False
    return True
if __name__ == "__main__":
    number = int(input("Enter a number: "))
    if is_prime(number):
        print(f"{number} is a prime number")
    else:
        print(f"{number} is not a prime number")
```

```
PS C:\Users\sriva\OneDrive\Desktop\3-2 academics\AI Assisted> & C:/Users/sriva/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/sriva/OneDrive/Deskto
p/3-2 academics/AI Assisted/lab2.py"
Enter a number: 5
5 is a prime number
PS C:\Users\sriva\OneDrive\Desktop\3-2 academics\AI Assisted> & C:/Users/sriva/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/sriva/OneDrive/Deskto
p/3-2 academics/AI Assisted/lab2.py"
Enter a number: 4
4 is not a prime number
PS C:\Users\sriva\OneDrive\Desktop\3-2 academics\AI Assisted>
```

EXPLANATION:

- This program defines a function is_prime() that checks whether a number is prime by trying to divide it by all numbers from 2 up to the square root of the number.
- If any division gives remainder 0, the function returns False (not prime). If no divisor is found, it returns True (prime).
- The main part of the program takes a number from the user, calls the function, and prints whether the number is prime or not.

❖ TASK-3:

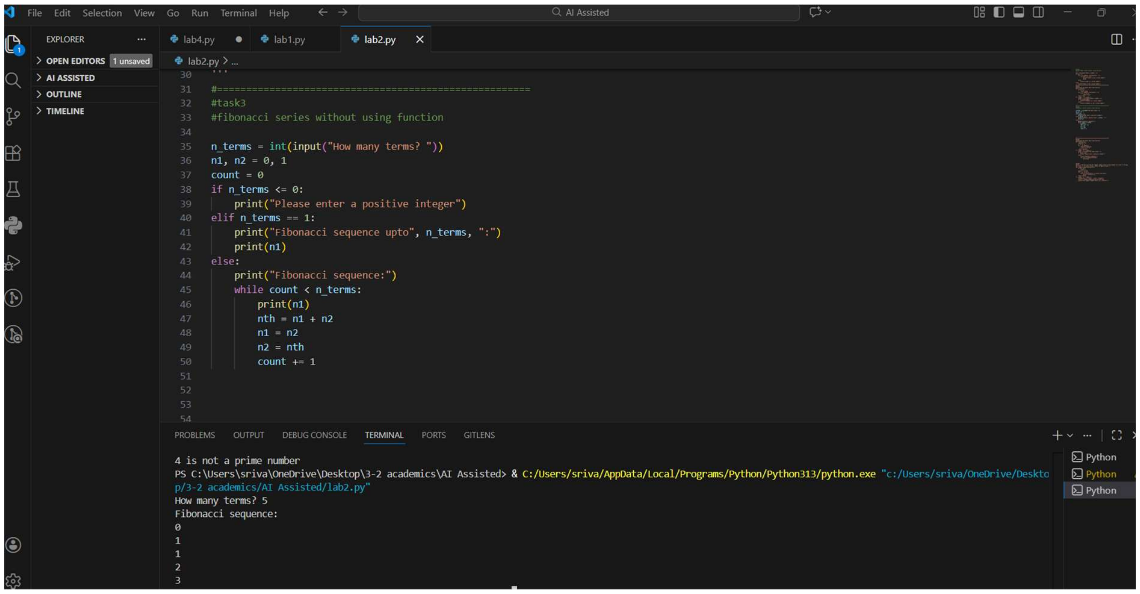
PROMPT:

Fibonacci series without using function.

CODE:

```
n_terms = int(input("How many terms? "))

n1, n2 = 0, 1

count = 0

if n_terms <= 0:

   print("Please enter a positive integer")

elif n_terms == 1:
   print("Fibonacci sequence upto", n_terms, ":")
   print(n1)
else:
   print("Fibonacci sequence:")
   while count < n_terms:
       print(n1)
       nth = n1 + n2
       n1 = n2
       n2 = nth
       count += 1
```

OUTPUT:



```
'''
#=======================================================
#task3
#fibonacci series without using function

n_terms = int(input("How many terms? "))
n1, n2 = 0, 1
count = 0
if n_terms <= 0:
    print("Please enter a positive integer")
elif n_terms == 1:
    print("Fibonacci sequence upto", n_terms, ":")
    print(n1)
else:
    print("Fibonacci sequence:")
    while count < n_terms:
        print(n1)
        nth = n1 + n2
        n1 = n2
        n2 = nth
        count += 1
```

```
4 is not a prime number
PS C:\Users\sriva\OneDrive\Desktop\3-2 academics\AI Assisted> & C:/Users/sriva/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/sriva/OneDrive/Deskto
p/3-2 academics/AI Assisted/lab2.py"
How many terms? 5
Fibonacci sequence:
0
1
1
2
3
```

EXPLANATION:

- This program prints the Fibonacci sequence up to the number of terms entered by the user.
  It starts with 0 and 1, then each next number is formed by adding the previous two numbers.
- If the user enters 0 or a negative number, it shows an error message.
  If the user enters 1, it prints only the first term (0).
  Otherwise, it uses a loop to generate and print the required Fibonacci numbers.

❖ TASK-4:

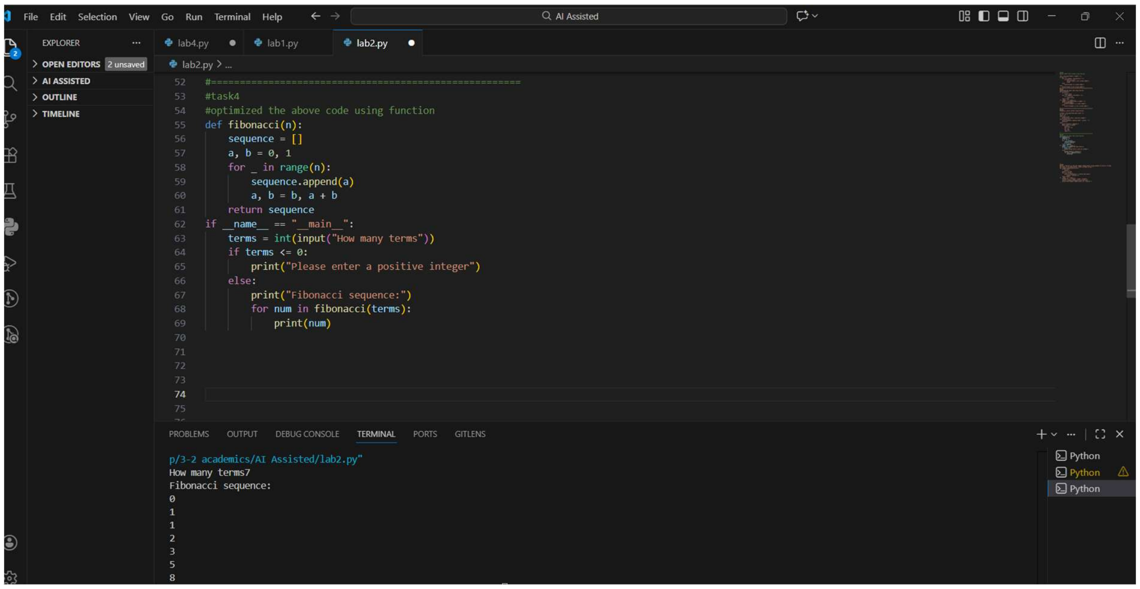
PROMPT:

optimized the above code using function.

CODE:

```
def fibonacci(n):
    sequence = []
    a, b = 0, 1
    for _ in range(n):
        sequence.append(a)
        a, b = b, a + b
    return sequence
if __name__ == "__main__":
    terms = int(input("How many terms"))
    if terms <= 0:
        print("Please enter a positive integer")
    else:
        print("Fibonacci sequence:")
        for num in fibonacci(terms):
            print(num)
```

OUTPUT:



```python
#==========================================================
#task4
#optimized the above code using function
def fibonacci(n):
    sequence = []
    a, b = 0, 1
    for _ in range(n):
        sequence.append(a)
        a, b = b, a + b
    return sequence
if __name__ == "__main__":
    terms = int(input("How many terms"))
    if terms <= 0:
        print("Please enter a positive integer")
    else:
        print("Fibonacci sequence:")
        for num in fibonacci(terms):
            print(num)
```

```
p/3-2 academics/AI Assisted/lab2.py"
How many terms7
Fibonacci sequence:
0
1
1
2
3
5
8
```

EXPLANATION:

- The function fibonacci(n) creates a list and fills it with Fibonacci numbers starting from 0 and 1. Each new number is formed by adding the previous two numbers. It returns the list of generated numbers.
- In the main part of the program, the user enters how many terms they want. If the number is 0 or negative, the program asks for a positive integer.
- Otherwise, it calls the fibonacci() function and prints the Fibonacci sequence.

❖ TASK-5:

PROMPT:

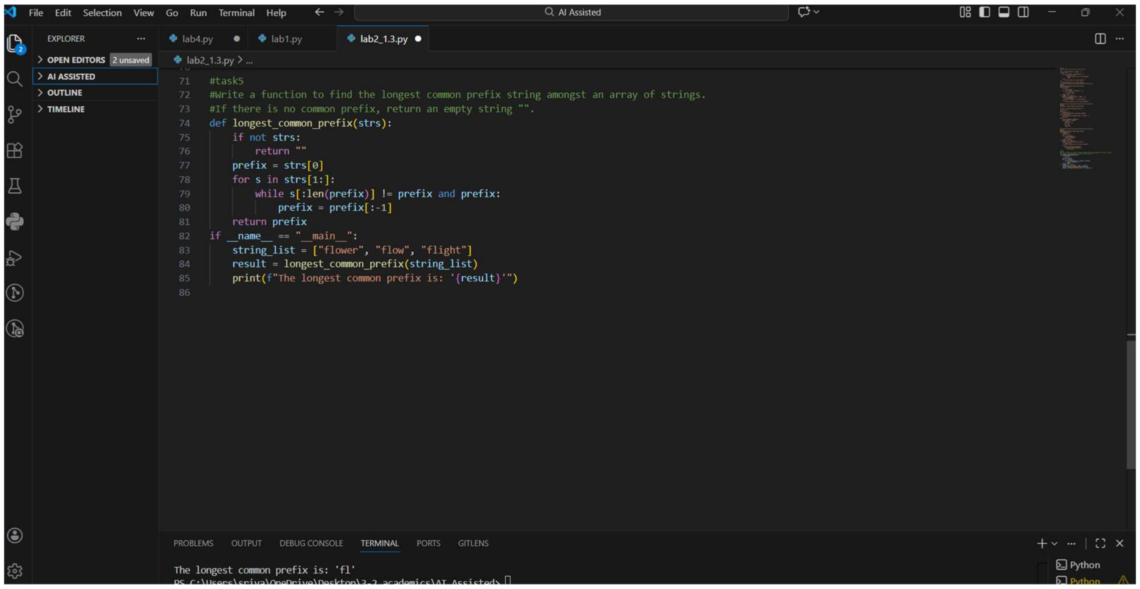
#Write a function to find the longest common prefix string amongst an array of strings.

#If there is no common prefix, return an empty string "".

CODE:

```python
def longest_common_prefix(strs):
  if not strs:
    return ""
  prefix = strs[0]
  for s in strs[1:]:
    while s[:len(prefix)] != prefix and prefix:
        prefix = prefix[:-1]
  return prefix
if __name__ == "__main__":
  string_list = ["flower", "flow", "flight"]
  result = longest_common_prefix(string_list)
  print(f"The longest common prefix is: '{result}'")
```

OUTPUT:



```python
#task5
#Write a function to find the longest common prefix string amongst an array of strings.
#If there is no common prefix, return an empty string "".
def longest_common_prefix(strs):
    if not strs:
        return ""
    prefix = strs[0]
    for s in strs[1:]:
        while s[:len(prefix)] != prefix and prefix:
            prefix = prefix[:-1]
    return prefix
if __name__ == "__main__":
    string_list = ["flower", "flow", "flight"]
    result = longest_common_prefix(string_list)
    print(f"The longest common prefix is: '{result}'")
```

```
The longest common prefix is: 'fl'
PS C:\Users\sriva\OneDrive\Desktop\3-2 academics\AI Assisted>
```

EXPLANATION:

- This program finds the longest common prefix (starting part) shared by all strings in a list.
- It begins by assuming the first string is the prefix. Then it compares this prefix with each remaining string. If the current prefix does not match the beginning of a string, it shortens the prefix by removing the last character and checks again.
- This continues until a match is found or the prefix becomes empty.
- After checking all strings, the remaining prefix is the longest common prefix, and it is printed as the result.