

## Assignment-1.4

2303A51343  
B-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

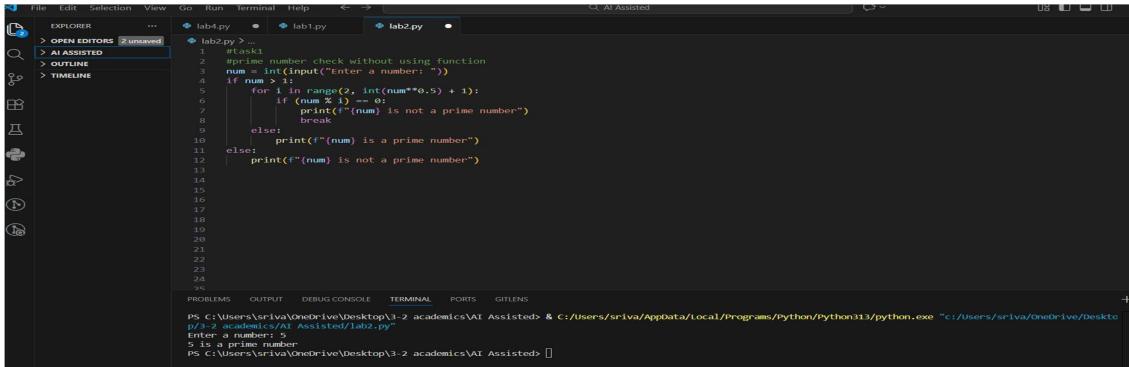
number") break

        else:

            print(f"{num} is a prime number") else:

            print(f"{num} is not a prime number")
```

#### **Output:**



The screenshot shows a Python code editor interface with the following details:

- File Explorer:** Shows three files: lab2.py, lab1.py, and lab2.py (the active file).
- Code Editor:** Displays the Python code for prime number checking.
- Terminal:** Shows the command-line output of the program's execution.
- Output:** The terminal output shows the user entering the number 5 and the program responding that it is a prime number.

#### **Analysis:**

- This program checks whether a number is prime by testing if it has any divisors from 2 to  $\sqrt{n}$ .
- 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

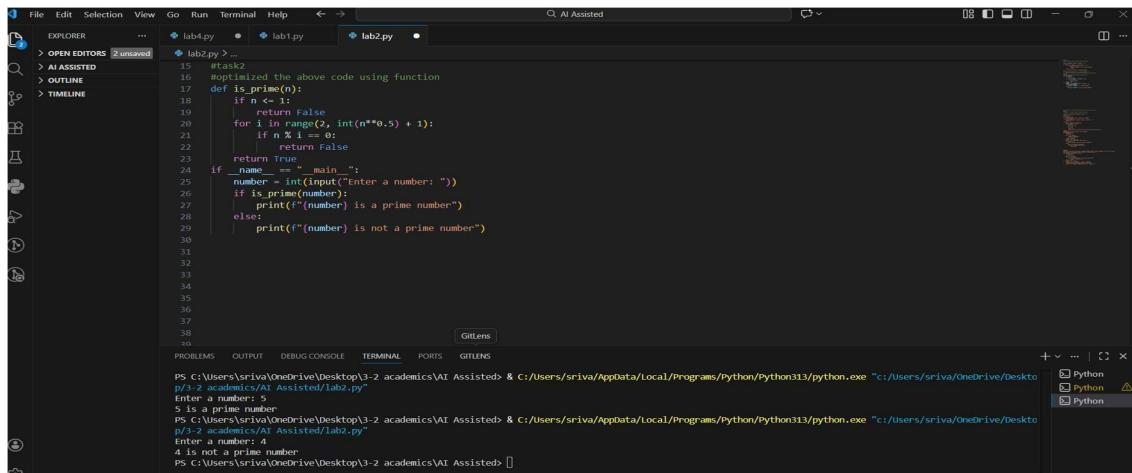
### Prompt:

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:



The screenshot shows the Visual Studio Code interface. In the center, there is a code editor window displaying Python code. The code defines a function `is_prime` that checks if a number is prime by iterating from 2 to the square root of the number. It also includes a main block that reads a number from the user and prints whether it is prime or not. To the right of the code editor, there is a terminal window showing the execution of the code and its output. The terminal shows the command `python lab2.py` being run, followed by the prompt "Enter a number:", the input "5", and the output "5 is a prime number".

### Analysis:

- 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:**

```

File Edit Selection View Go Run Terminal Help < -> AI Assisted
OPEN EDITORS 3 unsaved
EXPLORER ... lab4.py lab1.py lab2.py
AI ASSISTED
OUTLINE
TIMELINE

31 #=====
32 #task3
33 #fibonacci series without using function
34
35 n_terms = int(input("How many terms? "))
36 n1, n2 = 0, 1
37 count = 0
38 if n_terms <= 0:
39     print("Please enter a positive integer")
40 else:
41     print("Fibonacci sequence upto", n_terms, ":")
42     print(n1)
43     else:
44         print("Fibonacci sequence:")
45         while count < n_terms:
46             print(n1)
47             nth = n1 + n2
48             n1 = n2
49             n2 = nth
50             count += 1
51
52
53
54

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS GITLENS
+ ... | <>
Python Python Python

```

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

## Analysis:

- 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"))

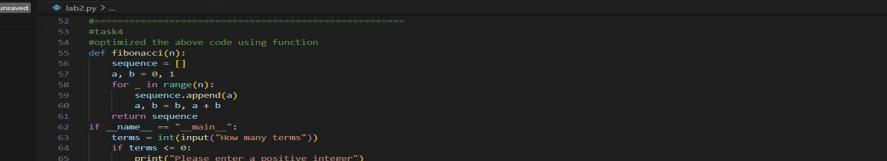
if terms <= 0:
    print("Please enter a positive integer")

else:

```

```
print("Fibonacci sequence:")  
  
for num in fibonacci(terms):  
    print(num)
```

## Output:



The screenshot shows a Visual Studio Code (VS Code) interface with the following details:

- File Explorer:** Shows files lab1.py, lab2.py, and lab2.py > ...
- Editor:** Displays Python code for generating a Fibonacci sequence. The code defines a function `fibonacci(n)` that generates a sequence of `n` terms. It uses a loop to calculate each term as the sum of the previous two. The main part of the script asks the user for input and prints the sequence.
- Terminal:** Shows the command `p/3-2 academics/AI Assisted/lab2.py` and the output of the Fibonacci sequence from 0 to 8.
- Status Bar:** Shows "AI Assisted".
- Bottom Status Bar:** Shows icons for Problems, Output, Debug Console, Terminal, Ports, GitLens, and Python environments.

### **Analysis:**

- 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.

## 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: def

```
longest_common_prefix(strs):    if  
not strs:          return ""  
  
prefix = strs[0]  
  
for s in strs[1:]:  
    while s[:len(prefix)] != 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:

The longest common prefix is: fl

### Analysis:

- 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.