UNIVERSITY OF CALOOCAN CITY
COMPUTER ENGINEERING DEPARTMENT

Data Structure and Algorithm

Laboratory Activity No. 6

Singly Linked Lists

Submitted by:
Catahan, Joshua A.

Instructor:
Engr. Maria Rizette H. Sayo

08, 23, 2025

I. Objectives

Introduction

A linked list is an organization of a list where each item in the list is in a separate node. Linked lists look like the links in a chain. Each link is attached to the next link by a reference that points to the next link in the chain. When working with a linked list, each link in the chain is called a Node. Each node consists of two pieces of information, an item, which is the data associated with the node, and a link to the next node in the linked list, often called next.

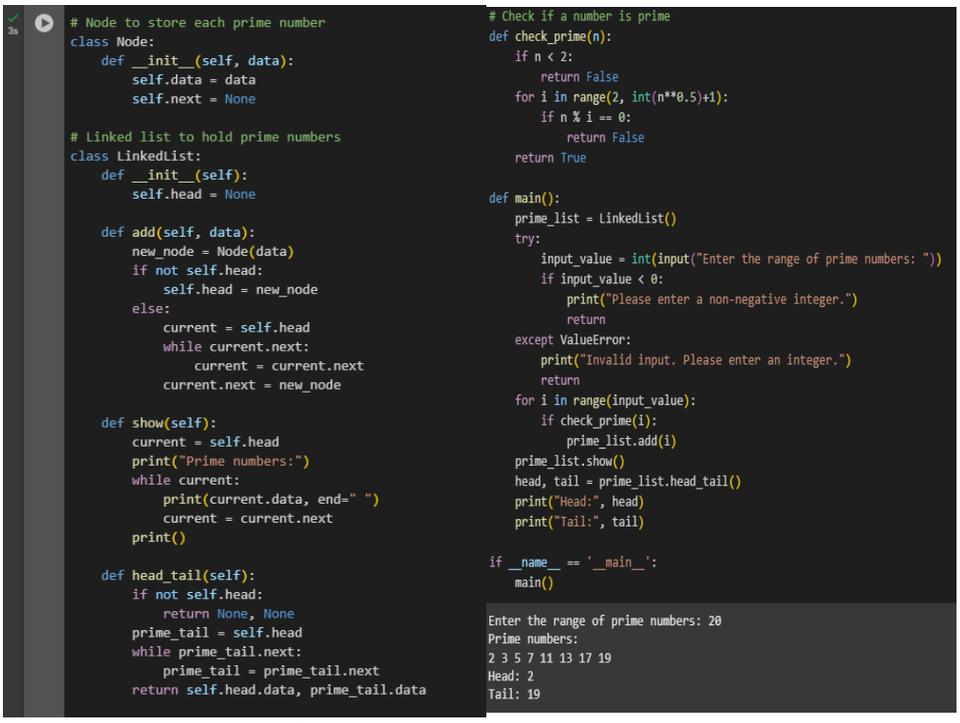
This laboratory activity aims to implement the principles and techniques in:

- Writing algorithms using Linked list
- Writing a python program that will perform the common operations in a singly linked list

II. Methods

- Write a Python program to create a singly linked list of prime numbers less than 20. By iterating through the list, display all the prime numbers, the head, and the tail of the list. (using Google Colab)
- Save your source codes to GitHub

III. Results



```python
# Node to store each prime number
class Node:
    def __init__(self, data):
        self.data = data
        self.next = None

# Linked list to hold prime numbers
class LinkedList:
    def __init__(self):
        self.head = None

    def add(self, data):
        new_node = Node(data)
        if not self.head:
            self.head = new_node
        else:
            current = self.head
            while current.next:
                current = current.next
            current.next = new_node

    def show(self):
        current = self.head
        print("Prime numbers:")
        while current:
            print(current.data, end=" ")
            current = current.next
        print()

    def head_tail(self):
        if not self.head:
            return None, None
        prime_tail = self.head
        while prime_tail.next:
            prime_tail = prime_tail.next
        return self.head.data, prime_tail.data
```

```python
# Check if a number is prime
def check_prime(n):
    if n < 2:
        return False
    for i in range(2, int(n**0.5)+1):
        if n % i == 0:
            return False
    return True

def main():
    prime_list = LinkedList()
    try:
        input_value = int(input("Enter the range of prime numbers: "))
        if input_value < 0:
            print("Please enter a non-negative integer.")
            return
    except ValueError:
        print("Invalid input. Please enter an integer.")
        return
    for i in range(input_value):
        if check_prime(i):
            prime_list.add(i)
    prime_list.show()
    head, tail = prime_list.head_tail()
    print("Head:", head)
    print("Tail:", tail)

if __name__ == '__main__':
    main()
```

```
Enter the range of prime numbers: 20
Prime numbers:
2 3 5 7 11 13 17 19
Head: 2
Tail: 19
```

Figure 1: Screenshot of the program

In this code as shown in figure 1, I created a program that uses a linked list to store and display prime numbers. I started by making a Node class, which holds the data for each prime number and a link to the next node. Then I built a LinkedList class with functions to add new primes, display all stored primes, and find the head and tail values of the list. To determine which numbers are prime, I wrote a check_prime function that checks divisibility from 2 up to the square root of the number. In the main function, the program asks the user for a range and validates the input. It then loops through all numbers in that range, uses the prime-checking function, and adds any prime numbers to the linked list. Finally, the program prints all the primes, along with the first (head) and last (tail) prime values. This way, I was able to practice combining data structures with number theory to create a simple but useful program.

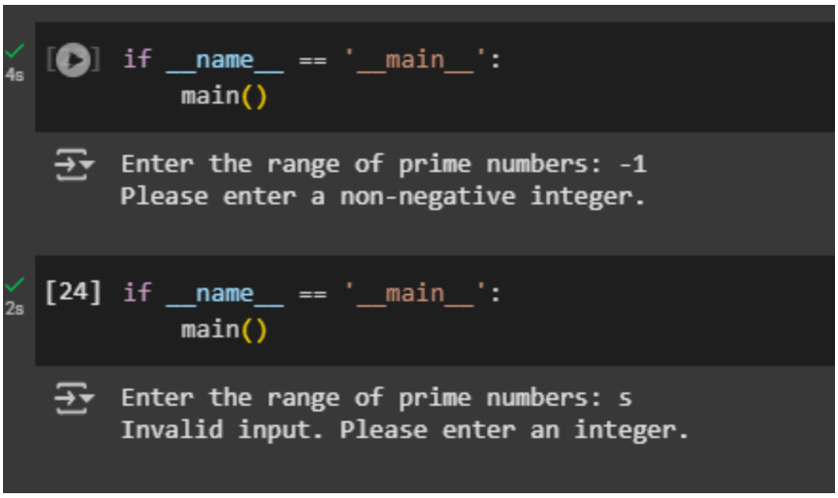


Figure 2: Screenshot of the program

In figure 2, I tried entering a negative integer and a character to show what would happen in the output.

IV. Conclusion

In summary, this laboratory activity combines the concept of linked lists with prime number generation. By storing primes in nodes, I was able to practice both data structures and basic algorithms in one project. Overall, the code shows how linked lists can be applied in solving simple mathematical problems.

References

[1] Co Arthur O.. “University of Caloocan City Computer Engineering Department Honor Code,” UCC-CpE Departmental Policies, 2020.