UNIVERSITY OF CALOOCAN CITY
COMPUTER ENGINEERING DEPARTMENT

Data Structure and Algorithm

Laboratory Activity No. 6

Singly Linked Lists

Submitted by:
Directo, Hannah Thea B.

Instructor:
Engr. Maria Rizette H. Sayo

August 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

```
class Node:
    def __init__(self, data):
        self.data = data
        self.next = None

class LinkedList:
    def __init__(self):
        self.head = None
        self.tail = None

    def prime_num(self, 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 append_if_prime(self, data):
        if not self.prime_num(data):
            return

        new_node = Node(data)
        if not self.head:
            self.head = self.tail = new_node
        else:
            self.tail.next = new_node
            self.tail = new_node
```

```
    def display(self):
        current = self.head
        print("Prime numbers from 1 to 20:")
        while current:
            print(current.data, end=' ')
            current = current.next
        print()

        if self.head:
            print(f"\nHead: {self.head.data}")
        else:
            print("Head: None")

        if self.tail:
            print(f"Tail: {self.tail.data}")
        else:
            print("Tail: None")


linked_list = LinkedList()
for i in range(1, 21):
    linked_list.append_if_prime(i)

linked_list.display()
```

```
Prime numbers from 1 to 20:
2 3 5 7 11 13 17 19

Head: 2
Tail: 19
```

Figure 1 Screenshot of program

Insights: This Python code creates a linked list that stores only the prime numbers between 1 and 20. It uses two classes: Node for individual elements and LinkedList for managing the list structure. The is prime method checks if a number is prime, and the append_if_prime method adds the number to the list only if it passes the prime check. The list keeps track of both the head and tail nodes for efficient appending. Finally, the display method prints all stored prime numbers and shows the values at the head and tail of the list, demonstrating basic linked list operations and conditional data storage.

IV. Conclusion

In this laboratory activity, we learned how to create and work with singly linked lists using Python. We wrote a program that stores prime numbers less than 20 in a linked list. This helped us understand how each node in a linked list holds data and a link to the next node. We also saw the importance of using a head and tail pointer to keep track of the start and end of the list. The program showed how to check if a number is prime before adding it to the list, which taught us how to combine conditions with data structures. By displaying the prime numbers along with the head and tail, we practiced how to navigate through a linked list. Overall, this activity improved our skills in organizing data and writing clear, efficient code for basic linked list operations.