UNIVERSITY OF CALOOCAN CITY
COMPUTER ENGINEERING DEPARTMENT

Data Structure and Algorithm

Laboratory Activity No. 6

Singly Linked Lists

Submitted by:
Talagtag, Mark Angel T.

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

```
        self.next = None

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

    def append(self, data):
        new_node = Node(data)
        if not self.head:
            self.head = new_node
        else:
            self.tail.next = new_node
        self.tail = new_node

    def display(self):
        current = self.head
        while current:
            print(current.data, end=" , ")
            current = current.next
        print("None")

def is_prime(num):
    if num < 2:
        return False
    for i in range(2, int(num**0.5) + 1):
        if num % i == 0:
            return False
    return True

# Create a SINGLY link list of prime Numbers less than 20
prime_list = SinglyLinkedList()
for number in range(20):
    if is_prime(number):
        prime_list.append(number)

# Display the list and head/tail
prime_list.display()
print(f"Head: {prime_list.head.data if prime_list.head else None}")
print(f"Tail: {prime_list.tail.data if prime_list.tail else None}")
```

```
2 , 3 , 5 , 7 , 11 , 13 , 17 , 19 , None
Head: 2
Tail: 19
```

ALGORITHM

1: Start

2: Define a Node Structure

3: Define the Singly Linked List Structure

4: Define a Prime Check Function

5: Build the Prime Linked List

6: Display the Results

IV. Conclusion

In this laboratory session, I gained valuable experience and understanding of how data structures specifically singly linked lists are implemented and used in programming. Prior to this, I was more familiar with basic structures like arrays, which have fixed sizes and direct index access. However, this lab introduced me to the concept of nodes and the dynamic nature of linked lists, which can grow or shrink in size without the need for memory reallocation.

Each node in a linked list contains two essential components: the data it holds and a pointer (or reference) to the next node. I learned how to construct these nodes using a simple class structure in Python and connect them in a sequence to form a singly linked list. Unlike arrays, linked lists do not store elements in contiguous memory locations; instead, each element points to the next, making insertion and deletion operations more efficient in certain cases.

One of the key tasks in the lab was to build a linked list of prime numbers less than 20. This required not only constructing the list but also implementing logic to determine whether a number is prime. Through this, I strengthened my understanding of both algorithmic thinking and data structure manipulation. I learned how to append new elements to the list, how to traverse the list to display its contents, and how to access both the head (first node) and tail (last node) of the list.

References

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