PROGRESS REPORT #1	
Course Code: CPE 201L	**Program:** BSCpE 2A
Course Title: Data Structure and Algorithm	**Date Performed:** September 13, 2025
Section: 2A	**Date Submitted:** September 13, 2025
Group Members: **Leader:** Villanueva, Bryan O. **Members:** Asugas, Kenneth R. Regondola, Jezreel P. Ruperto, April Anne A. Villacin, Justine R.	**Instructor:** Engr. Maria Rizette H. Sayo

1.Objectives

This project aims to implement the following:

- To create a diary application using doubly linked list.
- To create a user friendly interface with minimal design and easy navigation.
- To allow the users to write, edit, and delete daily diary entries.
- Provide users with an easy way to view and track their entries through a calendar.
- Ensure the user data is private and secure.
- Ensure the app works even without an internet connection.

2. Discussion

In this project, we designed a basic Diary Application in Python that is very much like the "Notes" application in smartphones. Rather than utilizing default lists or databases, we constructed the system with a Doubly Linked List (DLL) data type. Every entry in the diary is represented as a node, and the node has the content of the diary and pointers referring to both the previous and subsequent entries. Using a doubly linked list, we can implement insertion, deletion, traversal, and search operations of diary entries in an efficient manner. Traversal enables us to traverse the entries either forward (oldest to newest) or backward (newest to oldest), which proves to be highly beneficial for reading notes in chronological order. Insertion operations enable us to insert new diary entries dynamically without considering fixed array sizes, and deletion operations enable us to delete individual entries when required. This application illustrates the usefulness of linked lists in handling dynamic and sequential data. In contrast to arrays, which involve shifting the element positions during insertion and deletion, the doubly linked list arrangement allows these operations to be done more efficiently by updating node pointers. Thus, the diary application illustrates how data structures can be used to mimic real applications, like note taking or a journal system.

3. Materials and Equipment

- **PC** – Personal computer use to run programs, save files, and do tasks like typing, browsing, or coding.
- **Google Colab** – An online compiler where you can write and run python code.

4. Procedure

1. **Brainstorm ideas** – Think of different project topics and choose what we like best.
2. **Pick a project and type of data structure to use** – Decide the project goal and choose the right data structure (here, a Doubly Linked List).

3. **Analyze what functions to use for the diary** – Plan what actions the diary needs, like add, delete, or search entries.
4. **Create a program using Doubly Linked List** – Write the code using the planned functions to make the diary work.

5. Output

```
import json

class Node:
    def __init__(self,data):
        self.data = data
        self.date_created = datetime.now().strftime('%Y-%m-%d')
        self.prev = None
        self.next = None

class DoublyLinkedList:
    def __init__(self):
        self.head = None

    def append(self, data):
        new_node = Node(data)
        if not self.head:
            self.head = new_node
            return

        last_node = self.head
        while last_node.next:
            last_node = last_node.next

        last_node.next = new_node
        new_node.prev = last_node

    def display(self):
        current_node = self.head
        while current_node:
            print(current_node.data, end = " <=> ")
            current_node = current_node.next

        print("None")

    def delete(self, data):
      current_node = self.head
      while current_node:
        if current_node.prev is None:
          self.head = current_node.next
          if self.head:
            self.head.prev = None
        else:
          current_node.prev.next = current_node.next
          if current_node.next:
            current_node.next.prev = current_node.prev

        print(f"Entry {data} deleted.")
        return
      current_node = current_node.next
      print(f"Entry {data} not found.")

    def insert(self):
```

6. Conclusion

In conclusion, this project successfully demonstrated how a Doubly Linked List (DLL) can be applied to build a dynamic and functional Diary Application in Python. By structuring each diary entry as a node within the DLL, we were able to implement essential operations such as insertion, deletion, traversal, and search efficiently. The application highlights the practical relevance of data structures in real world scenarios, especially in managing sequential and frequently updated data. Features like forward and backward traversal allowed for intuitive chronological navigation of entries, while the flexible node manipulation ensured seamless addition and removal of notes. Moreover, the diary app was designed with key user focused goals, simplicity, offline functionality, data privacy, and ease of use. The result is a lightweight, intuitive interface that enables users to write, edit, and manage diary entries securely and conveniently even without an internet connection.

Lab Activity Rubric

Criteria	Ratings						Pts
SO 7 PI 1 **Student Outcome 7.1** Acquire and apply new knowledge from outside sources. threshold: 4.8 pts	6 pts Excellent | Educational interests and pursuits exist and flourish outside classroom requirements,knowledge and/or experiences are pursued independently and applies knowledge learned into practice	5 pts Good | Educational interests and pursuits exist and flourish outside classroom requirements,knowledge and/or experiences are pursued independently	4 pts Satisfactory | Look beyond classroom requirements, showing interest in pursuing knowledge independently	3 pts Unsatisfactory | Begins to look beyond classroom requirements, showing interest in pursuing knowledge independently	2 pts Poor | Relies on classroom instruction only	1 pts Very Poor | No initiative or interest in acquiring new knowledge	6 pts
SO 7 PI 2 **Student Outcome 7.2** Learn independently threshold: 4.8 pts	6 pts Excellent | Completes an assigned task independently and practices continuous improvement	5 pts Good | Completes an assigned task without supervision or guidance	4 pts Satisfactory | Requires minimal guidance to complete an assigned task	3 pts Unsatisfactory | Requires detailed or step-by-step instructions to complete a task	2 pts Poor | Shows little interest to complete a task independently	1 pts Very Poor | No interest to complete a task independently	6 pts
SO 7 PI 3 **Student Outcome 7.3** Critical thinking in the broadest context of technological change threshold: 4.8 pts	6 pts Excellent | Synthesizes and integrates information from a variety of sources; formulates a clear and precise perspective; draws appropriate conclusions	5 pts Good | Evaluate information from a variety of sources; formulates a clear and precise perspective.	4 pts Satisfactory | Analyze information from a variety of sources; formulates a clear and precise perspective.	3 pts Unsatisfactory | Apply the gathered information to formulate the problem	2 pts Poor | Gather and summarized the information from a variety of sources but failed to formulate the problem	1 pts Very Poor | Gather information from a variety of sources	6 pts
SO 7 PI 4 **Student Outcome 7.4** Creativity and adaptability to new and emerging technologies threshold: 4.8 pts	6 pts Excellent | Ideas are combined in original and creative ways in line with the new and emerging technology trends to solve a problem or address an issue.	5 pts Good | Ideas are creative and adapt the new knowledge to solve a problem or address an issue	4 pts Satisfactory | Ideas are creative in solving a problem, or address an issue	3 pts Unsatisfactory | Shows some creative ways to solve the problem	2 pts Poor | Shows initiative and attempt to develop creative ideas to solve the problem	1 pts Very Poor | Ideas are copied or restated from the sources consulted	6 pts
							Total Points: 24