**Nama :Ahmad wahyu abadi setiawan**

**Kelas : IX RPL**

**No absen : 02**

## **Jawaban Soal Teori**

1. Jelaskan perbedaan mendasar antara Array dan Linked List dalam hal penambahan data di tengah!

* Array memiliki ukuran tetap (terutama di bahasa seperti C atau Java), dan menyisipkan data di tengah membutuhkan penggeseran semua elemen setelahnya, sehingga lebih lambat untuk operasi sisip.
* Linked List memiliki ukuran dinamis dan tidak perlu menggeser data saat menyisipkan. Cukup ubah pointer (next) dari node sebelumnya untuk menunjuk ke node baru, dan node baru menunjuk ke node setelahnya.

2. Apa fungsi dari head dalam sebuah Linked List? Mengapa ia sangat penting?

* Head adalah referensi ke simpul pertama dalam Linked List.
* Head sangat penting karena tanpa head, kita tidak bisa mengakses seluruh list. Semua traversal atau operasi dimulai dari head.

**Studi Kasus Praktik**

class Node:

def \_\_init\_\_(self, data):

self.data = data

self.next = None

class LinkedList:

def \_\_init\_\_(self):

self.head = None

def print\_list(self):

print("Isi Playlist:")

current\_node = self.head

if current\_node is None:

print("(Playlist kosong)")

return

while current\_node is not None:

print(f"- {current\_node.data}")

current\_node = current\_node.next

def append(self, data):

new\_node = Node(data)

if self.head is None:

self.head = new\_node

return

last\_node = self.head

while last\_node.next:

last\_node = last\_node.next

last\_node.next = new\_node

def prepend(self, data):

new\_node = Node(data)

new\_node.next = self.head

self.head = new\_node

def delete\_node(self, key):

current\_node = self.head

if current\_node and current\_node.data == key:

self.head = current\_node.next

current\_node = None

return

prev = None

while current\_node and current\_node.data != key:

prev = current\_node

current\_node = current\_node.next

if current\_node is None:

print(f"Error: Data '{key}' tidak ditemukan.")

return

prev.next = current\_node.next

current\_node = None

def insert\_after(self, prev\_node\_data, new\_data):

current\_node = self.head

while current\_node:

if current\_node.data == prev\_node\_data:

new\_node = Node(new\_data)

new\_node.next = current\_node.next

current\_node.next = new\_node

print(f"Sukses: '{new\_data}' disisipkan setelah '{prev\_node\_data}'")

return

current\_node = current\_node.next

print(f"Error: Data pendahulu '{prev\_node\_data}' tidak ditemukan.")

playlist = LinkedList()

playlist.append("Lagu Pop - A")

playlist.append("Lagu Rock - B")

playlist.append("Lagu Jazz - C")

playlist.print\_list()

print("\nMenambah lagu intro...")

playlist.prepend("Intro Musik")

playlist.print\_list()

print("\nMenyisipkan 'Lagu Iklan' setelah 'Intro Musik'...")

playlist.insert\_after("Intro Musik", "Lagu Iklan")

playlist.print\_list()

print("\nMenghapus 'Lagu Rock - B'...")

playlist.delete\_node("Lagu Rock - B")

playlist.print\_list()

print("\nMencoba menghapus lagu yang salah...")

playlist.delete\_node("Lagu Dangdut")

playlist.print\_list()

print("\nMenambah lagu penutup...")

playlist.append("Outro Musik")

playlist.print\_list()

print("\nMembuat playlist baru dan mengeceknya...")

playlist\_kosong = LinkedList()

playlist\_kosong.print\_list()