**ID: hrajranj**

**Day 13 – 26th June 2025**

**Home Tasks:**

**1. List Advantages and disadvantages of linked List**

**Ans:**

| **Advantage** | **Explanation** |
| --- | --- |
| 🔁 Dynamic Size | You can grow/shrink the list at runtime (no need to define a fixed size like arrays). |
| 🧱 Efficient Insert/Delete | Insertions and deletions are faster (especially in the middle) since you just change links — no shifting like arrays. |
| 🧭 No Wasted Space | No need to pre-allocate memory — each node is created as needed. |
| ↔️ Easy to Implement Data Structures | Useful for implementing stacks, queues, graphs, and hash chains. |
| 🧠 Better memory management | Good for applications where memory allocation and deallocation happen frequently. |

| **Disadvantage** | **Explanation** |
| --- | --- |
| 🐌 Slower Access Time | No direct access — must traverse from the head to get to a node (unlike arrays). |
| 🧠 More Memory Usage | Each node stores a pointer/reference in addition to data (extra memory per node). |
| 🌀 Poor Cache Performance | Non-contiguous memory makes it less CPU-cache-friendly than arrays. |
| 🛠️ Complex Implementation | Requires careful handling of pointers, especially in insertion/deletion. |
| 🕳️ No Random Access | You cannot do list[4] directly — you must walk node by node. |

**2. Applications of Linked list**

**Ans:**

**Applications of Linked List**

1. Dynamic Data Structures

* Linked lists are used to implement stacks, queues, and deques.
* Unlike arrays, they allow dynamic memory allocation and efficient insert/delete operations.

2. Circular Lists (Schedulers)

* Circular linked lists are used in applications like:
  + Round-robin CPU scheduling
  + Multiplayer games (turn-based rotation)
  + Buffer rotation in OS or networking

3. Memory Management

* Operating systems use linked lists to manage:
  + Free memory blocks
  + Page tables
  + File allocation tables (FAT)

4. Adjacency Lists in Graphs

* Graphs are often represented using adjacency lists, which are implemented using linked lists for efficient traversal and memory use.

5. Undo/Redo Functionality

* Text editors, IDEs, and even drawing tools use linked lists to track changes (each node stores a state).

6. Polynomial Arithmetic

* Polynomial expressions can be efficiently stored and manipulated using linked lists where each node holds coefficient and exponent.

7. Hash Tables (with Chaining)

* Linked lists are used to resolve collisions in hash tables via chaining (each bucket has a linked list of entries).

8. File Systems

* Some older or custom file systems use linked allocation methods where files are stored in blocks linked together.

9. Skip Lists

* A special form of linked list with multiple levels used in high-performance search and database indexing.

10. Game Development

* Used to track and manage game entities like enemies, bullets, or events dynamically during gameplay.

Bonus:

Linked lists are a foundation for understanding:

* Pointers
* Dynamic memory
* Recursion
* Low-level data structure management