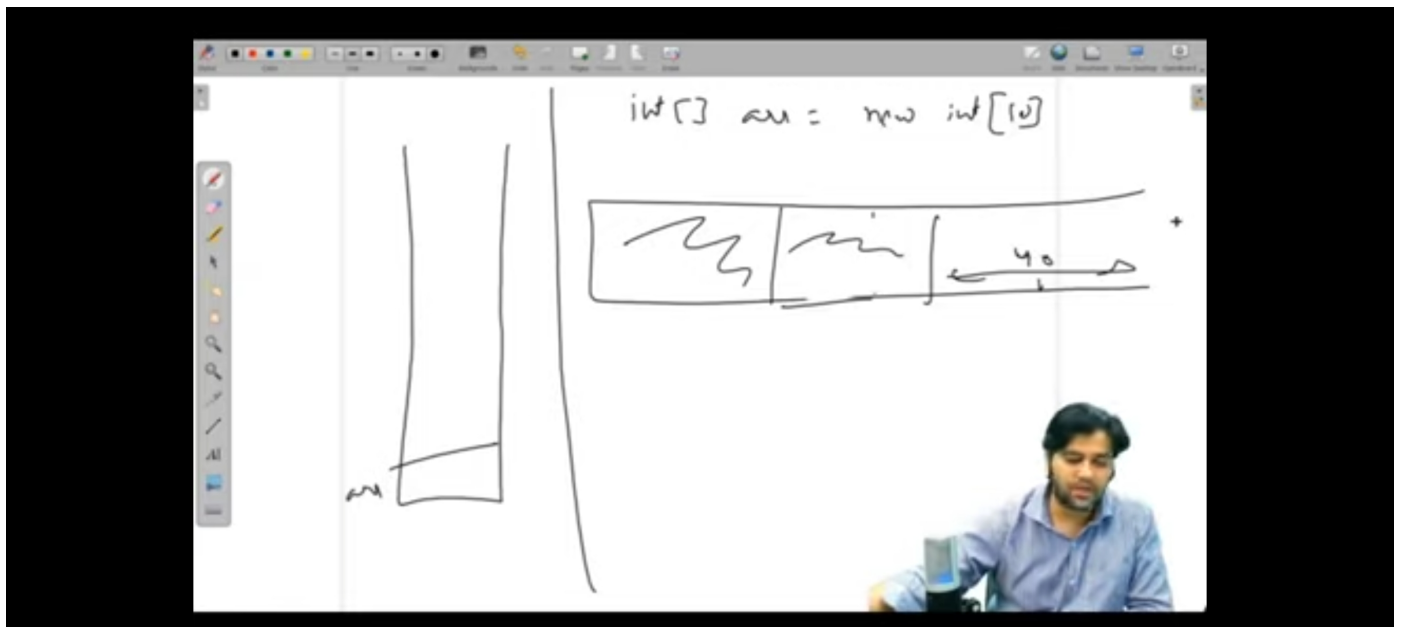
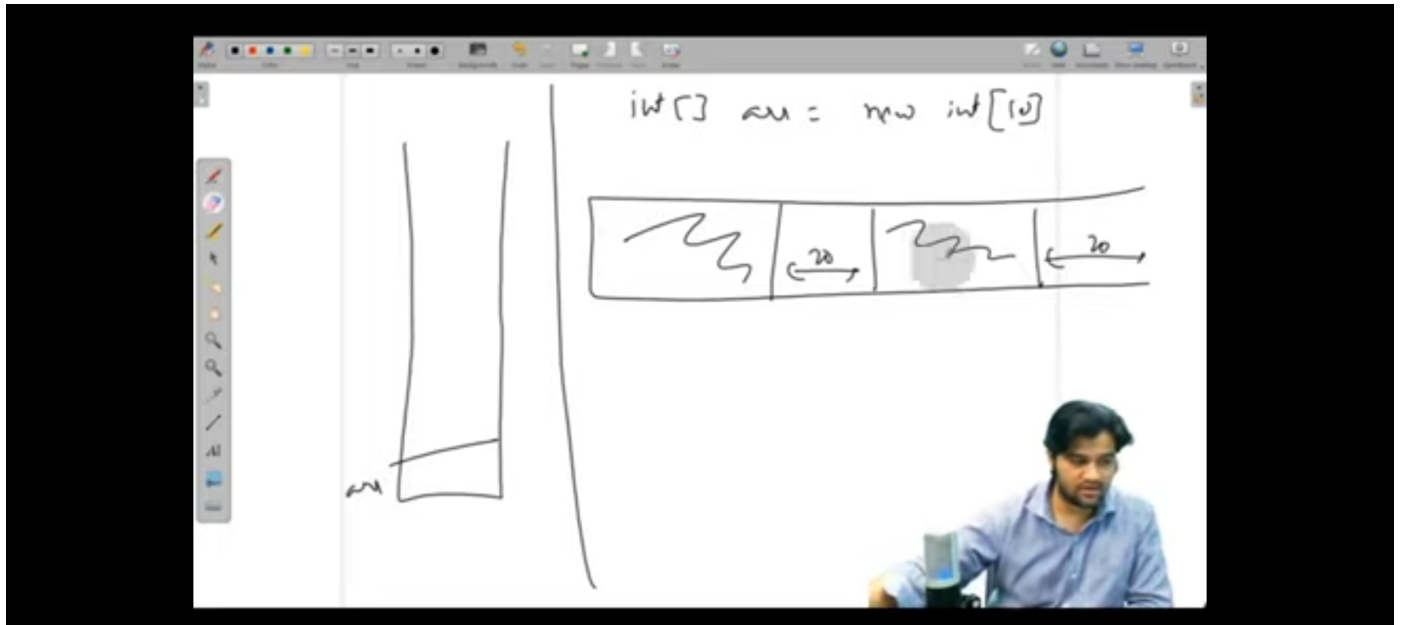


- LINKED LISTS
- QUESTIONS

# LINKED LISTS



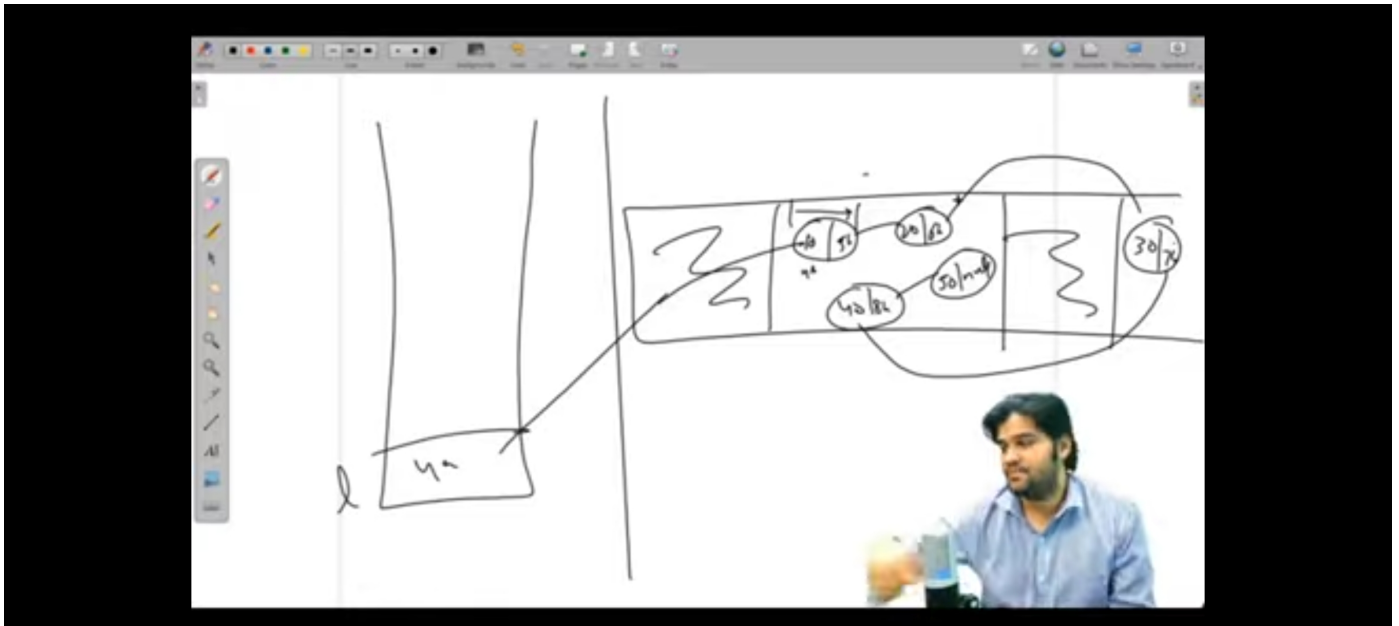
int[] arr = new int[10]

The diagram illustrates the memory layout for an array. A variable 'arr' is shown in a memory box, containing the address '4k'. An arrow points from this address to a specific location in a larger memory block. This block is divided into 10 slots, indexed from 0 to 9. The first slot contains a squiggle, and the second slot contains the text 'in'. The rest of the slots are empty.

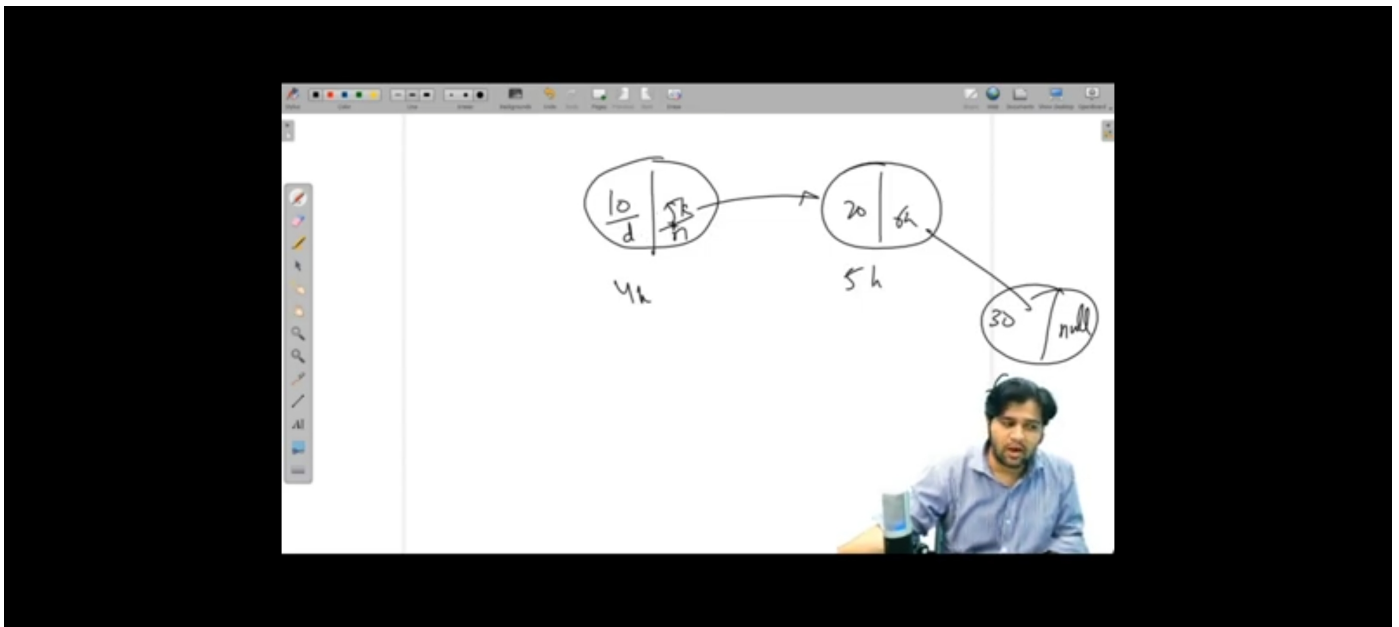
The diagram shows a linked list structure. A pointer variable 'l' points to the first node. Each node is represented as a rectangle divided into two parts. The left part contains a squiggle, and the right part contains a circle with a cross. Arrows connect the 'next' pointer of one node to the start of the next node, forming a sequence of three nodes.

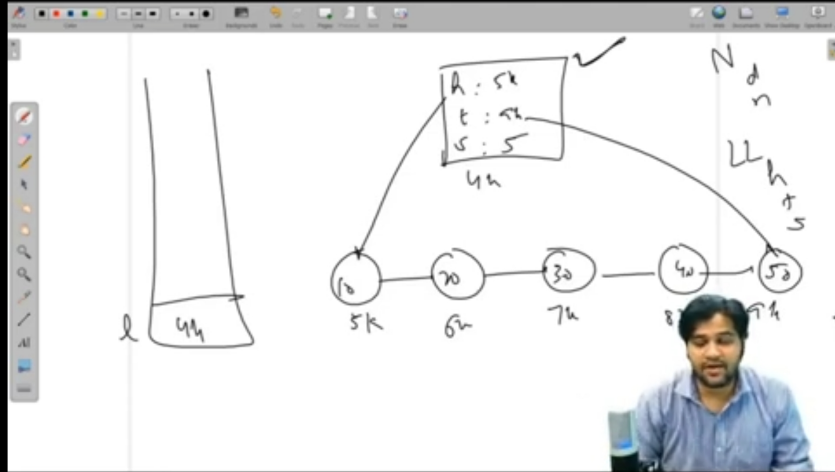
The diagram shows a graph structure. A pointer variable 'l' points to a node. The nodes are represented as circles divided into two parts. The left part contains a value followed by a slash and another value (e.g., '10/5k', '20/6k', '30/7k', '40/8k', '50/9k', '60/10k', '70/11k', '80/12k', '90/13k', '100/14k'). The right part contains a value followed by a slash and another value (e.g., '20/6k', '30/7k', '40/8k', '50/9k', '60/10k', '70/11k', '80/12k', '90/13k', '100/14k'). Arrows connect the nodes in a sequence, forming a path through the graph.

- Linked List utilized space in case of fragmented memory



- Array use 4-Bytes to store a integer value
- But Linked List use 8-Bytes to store a Integer value
  - 8-Bytes = 4bytes (value) + 4bytes (address)

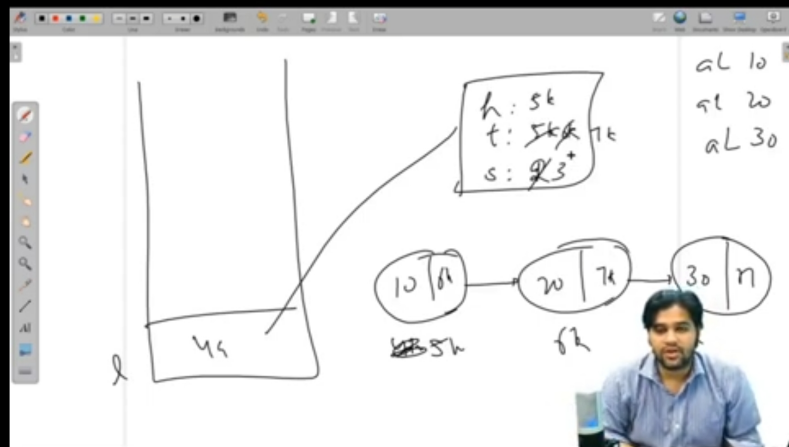




No connection

## QUESTIONS

### 1. Add Last in Linked List



2.