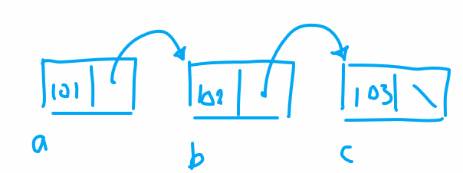
**March 8, 2021**

1. **Linked List**

**A data structure to represent a list**

**Main reason: we want the flexibility to add or remove one element from the list.**

****

**Struct Node**

**{**

**Int id;**

**Node \*next;**

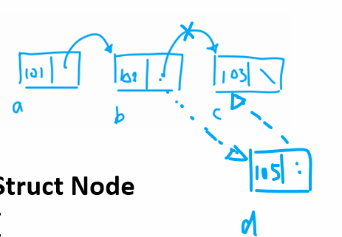
**};**

**Node a, b, c;**

**a.id=101, b.id=102, c.id=103;**

**a.next=&b, b.next=&c, c.next=NULL; // NULL=0**

**Node d; d.id=105;**

****

**b.next=&d, d.next=&c;**

**singly-linked list**

**doubly-linked list**

**Struct Node2**

**{**

**Int id;**

**Node2 \*next;**

**Node2 \*prev;**

**} a , b, c;**

**a.id=101, b.id=102, c.id=103;**

**a.next=&b, b.next=&c, c.next=NULL;**

**a.prev=NULL, b.prev=&a, c.prev=&b;**

1. **Dynamic Arrays**

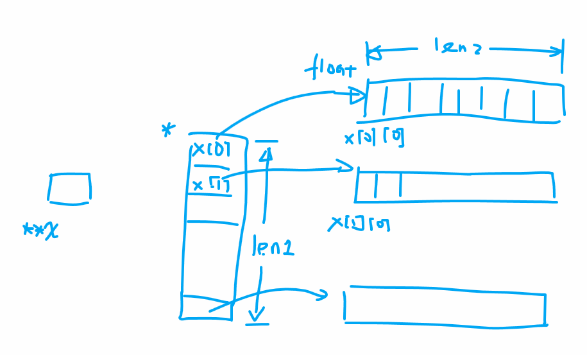
**Int len1, len2;**

**Cin >> len1 >> len2;**

**Float \*\*x = new float\*[len1];**

**For(int i=0; i<len1; i++)**

**X[i] = new float[len2];**

****

**// deallocation**

**For(int i=0; i<len1; i++)**

**Delete []x[i];**

**Delete []x;**

**We normally expect a double pointer variable is pointing to the beginning memory address of a single point variable.**

**Double \*\*x;**

**Double \*y;**

**x = &y;**