**DELETE(INFO, LINK, START, AVAIL, ITEM)**

**Step-by-step :**

1. **Initialize:**  
   Set PTR := START (start at the beginning of the list)  
   Set LOCP := NULL (no predecessor yet)
2. **Repeat while** PTR ≠ NULL:
   * If INFO[PTR] == ITEM, then:
     + Set LOC := PTR (item found)
     + Exit the loop (done searching)
   * Else:
     + Set LOCP := PTR (current becomes previous)
     + Set PTR := LINK[PTR] (move to next node)
3. **If loop ends without finding ITEM:**
   * Set LOC := NULL (item not found)
4. **Exit.**

2. If LOC = NULL, then:

Write: ITEM not in list, and Exit.

3. If LOCP = NULL, then:

Set START := LINK[START].

Else:

Set LINK[LOCP] := LINK[LOC].

1. Set LINK[LOC] := AVAIL

Set AVAIL := LOC

1. Exit.

## **Algorithm: Insert at Last (End of Linked List)**

**Input:**

* INFO[] and LINK[] arrays (representing the linked list)
* START pointer (beginning of the list)
* AVAIL pointer (beginning of free node list)
* ITEM = value to insert

**Output:**

* Updated linked list with ITEM added at the end

### ✅ ****Steps:****

1. **Check overflow**  
   If AVAIL == NULL, then:  
   → Print "OVERFLOW" and Exit
2. **Get new node from AVAIL list**  
   Set NEW := AVAIL  
   Set AVAIL := LINK[AVAIL]
3. **Store data**  
   Set INFO[NEW] := ITEM  
   Set LINK[NEW] := NULL
4. **If list is empty**  
   If START == NULL, then:  
   → Set START := NEW  
   → Exit
5. **Traverse to last node**  
   Set PTR := START  
   Repeat while LINK[PTR] ≠ NULL:  
   → Set PTR := LINK[PTR]
6. **Insert at end**  
   Set LINK[PTR] := NEW
7. **Exit**

### 📌 Example:

Suppose:

* INFO = [10, 20, 30]
* LINK = [2, 3, NULL]
* START = 1, AVAIL = 4
* ITEM = 40

After insertion:

* INFO = [10, 20, 30, 40]
* LINK = [2, 3, 4, NULL]
* START = 1

Final list: 10 → 20 → 30 → 40

## **🔧 Algorithm: INSERT(INFO, LINK, START, AVAIL, ITEM)**

(Insert ITEM in a sorted linked list)

### 📥 ****Input:****

* INFO[] — Array holding the data of each node
* LINK[] — Array holding the next node index
* START — Pointer to the beginning of the linked list
* AVAIL — Pointer to the beginning of the free node list
* ITEM — The new value to be inserted

### 📤 ****Output:****

* Updated INFO, LINK, and START arrays with ITEM inserted at the correct position to maintain **ascending order**

### ✅ ****Steps:****

1. **Check overflow:**  
   If AVAIL == NULL, then:  
   → Print "OVERFLOW" and Exit
2. **Allocate new node from AVAIL:**  
   Set NEW := AVAIL  
   Set AVAIL := LINK[AVAIL]  
   Set INFO[NEW] := ITEM
3. **Insert at beginning (if list is empty OR ITEM < first item):**  
   If START == NULL OR ITEM < INFO[START], then:  
   → Set LINK[NEW] := START  
   → Set START := NEW  
   → Exit
4. **Find position to insert:**  
   Set SAVE := START  
   Set PTR := LINK[START]  
   While PTR ≠ NULL AND INFO[PTR] < ITEM, do:  
   → SAVE := PTR  
   → PTR := LINK[PTR]
5. **Insert ITEM between SAVE and PTR:**  
   Set LINK[NEW] := PTR  
   Set LINK[SAVE] := NEW
6. **Exit**