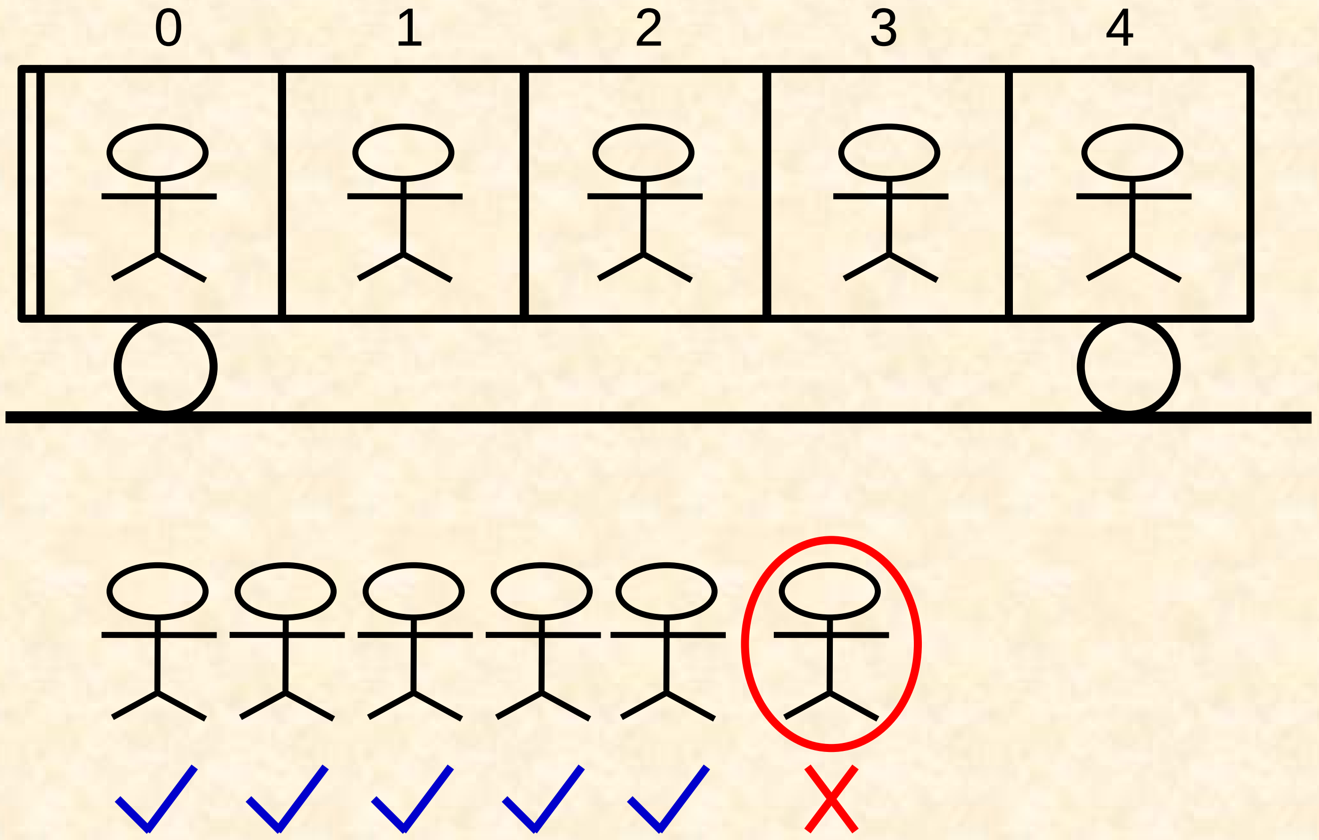
Linked List

by:

Bharat V. Chawda
Computer Engineering Department,
BBIT, VVNagar, Gujarat, India

Array

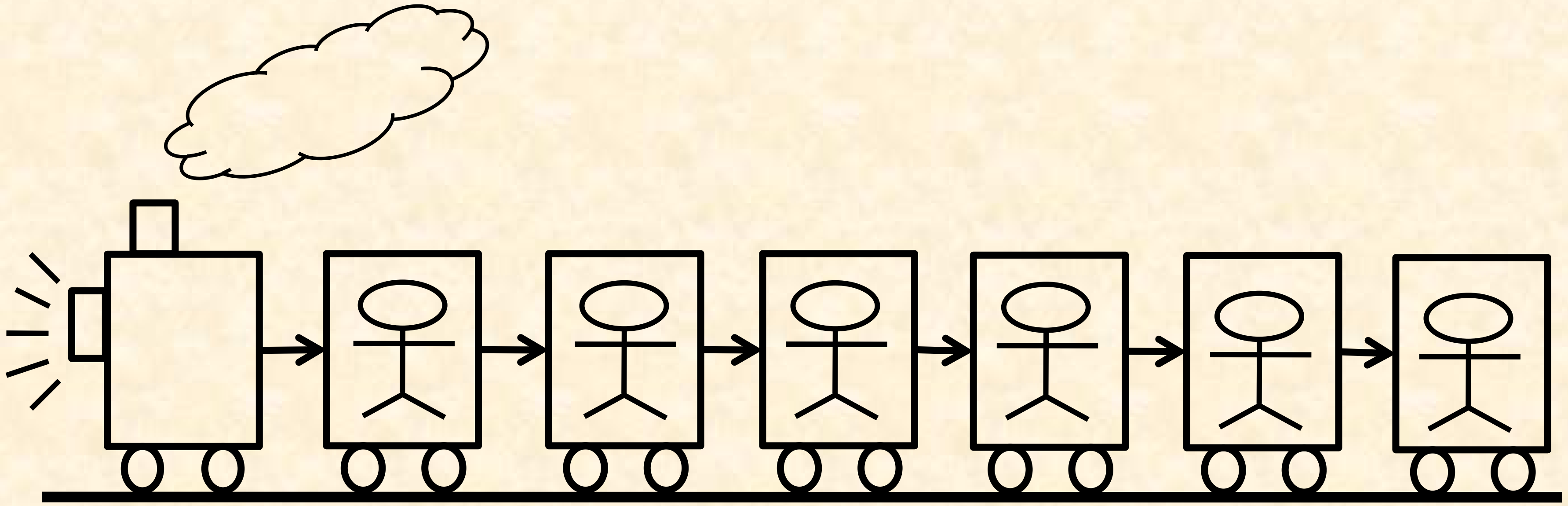
Array → Five Elements

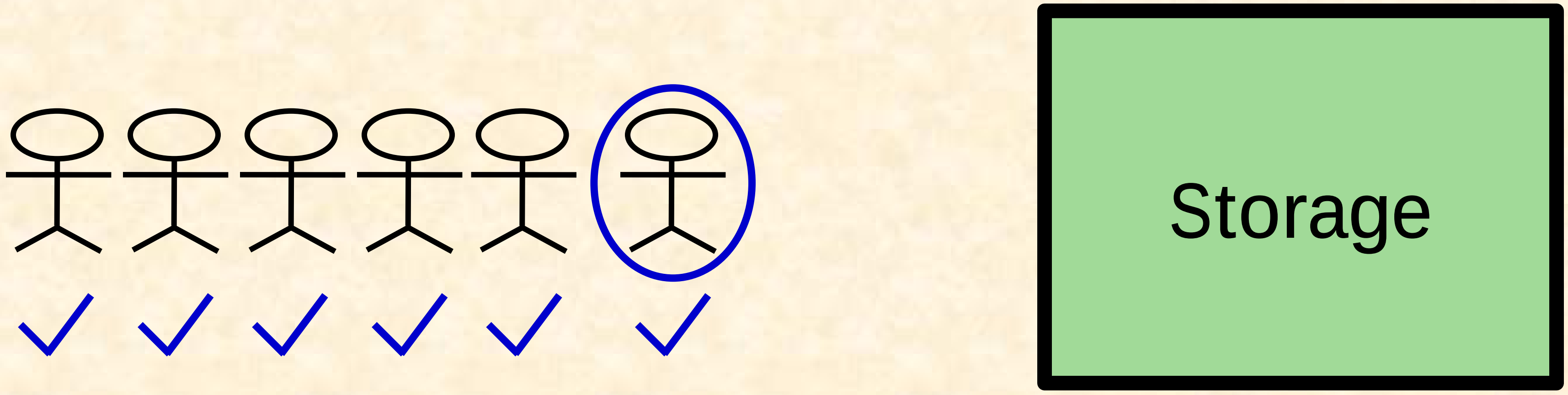


Array

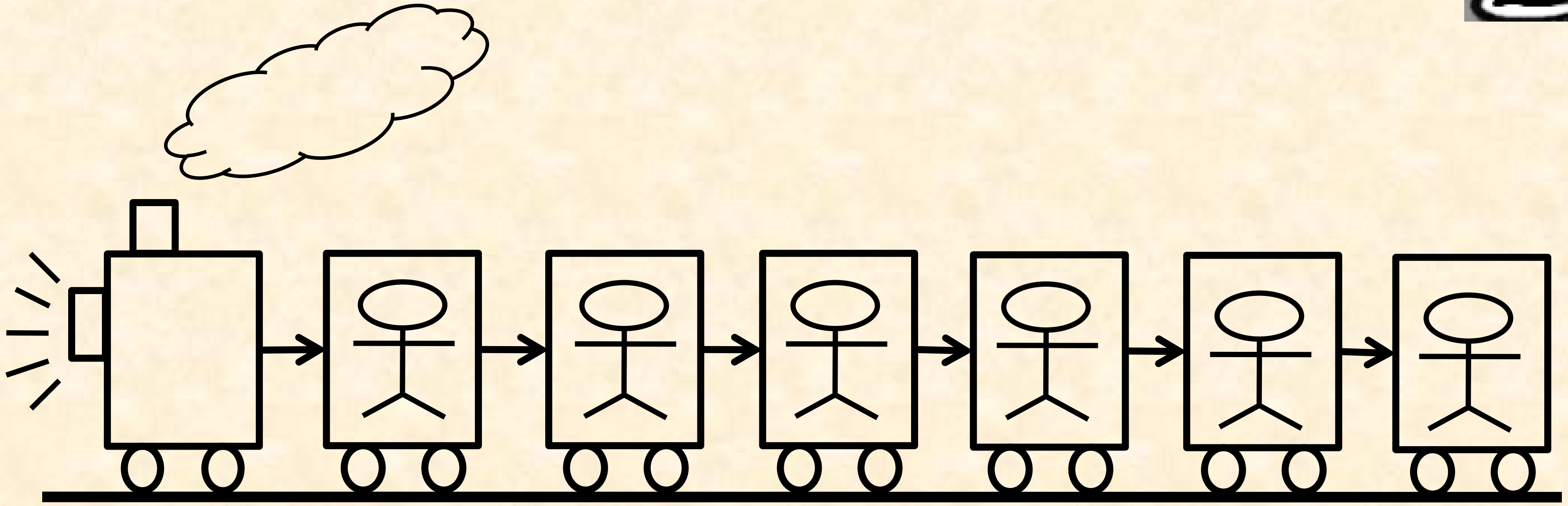
- Manage marks of 5 subjects:
  - int a [5];
  - What if, there are 6 subjects?
  - What if, there are only 3 subjects?
- Manage marks of 5 subjects for 10 students:
  - int a [10][5];
  - What if, there are 11 students?
  - What if, there are only 7 students?

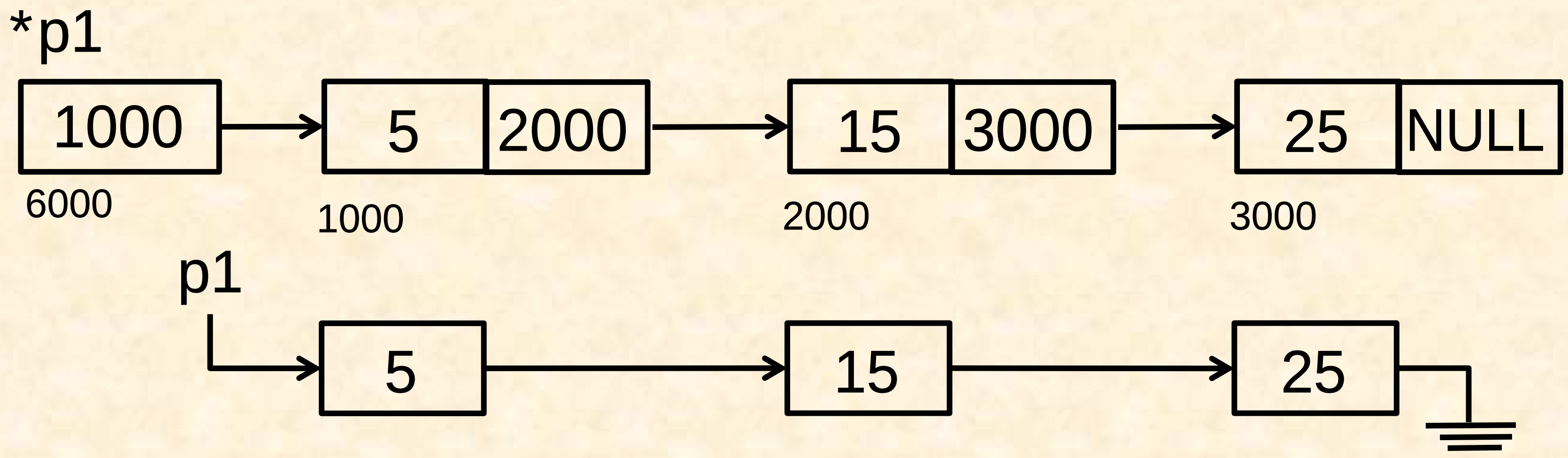
Linked List





Linked List

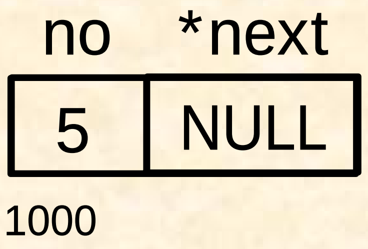




Linked List – What?

- **Def:**
  - “Collection of nodes in which –
  - each node **points to** another node in a **linear** sequence.”
  - Each node has **2** parts:

    Data/Information + Address/Pointer



Linked List - Classification

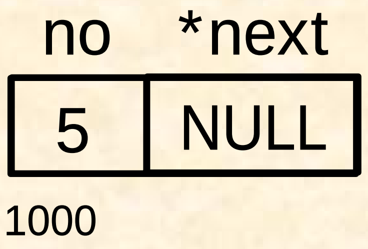
- **Types:**
  - Singly Linked List
  - Doubly Linked List
  - Circular Linked List (Singly or Doubly)

Singly Linked List

- **Def:**
  - "A linked list in which each node contains **one** pointer –

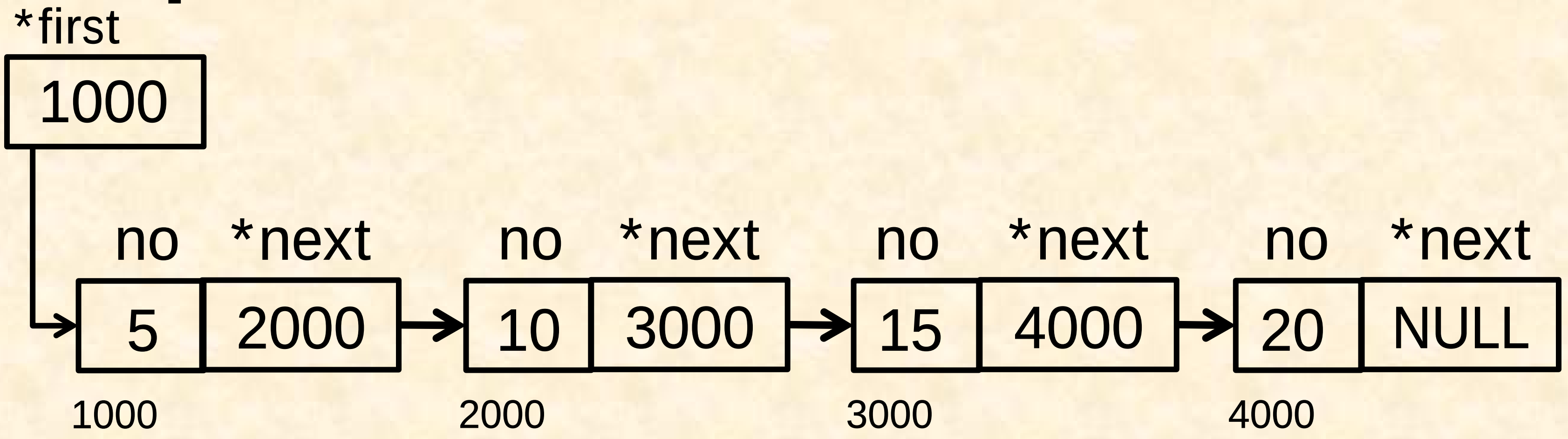
    - to point to its **next** node."
  - Each node has **2** fields:

    Information + Next Pointer



SLL: Representation

- **Representation:**



- **Structure:**

```
struct test
{
    int   no;
    struct   test   *next;
};
```

SLL: Operations

- **Insert**
  - Insert at End (Append)
  - Insert at Front
  - Insert After
  - Insert Before
- **Traverse**
- **Delete**
- **Search**
- **Count**

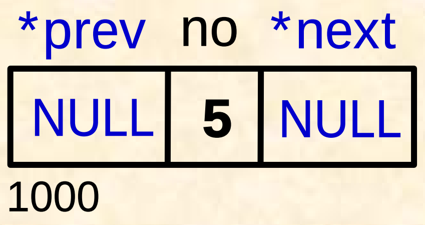
Doubly Linked List

- **Def:**
  - "A linked list in which each node contains **two** pointers –

    - one, to point to its **next** node,

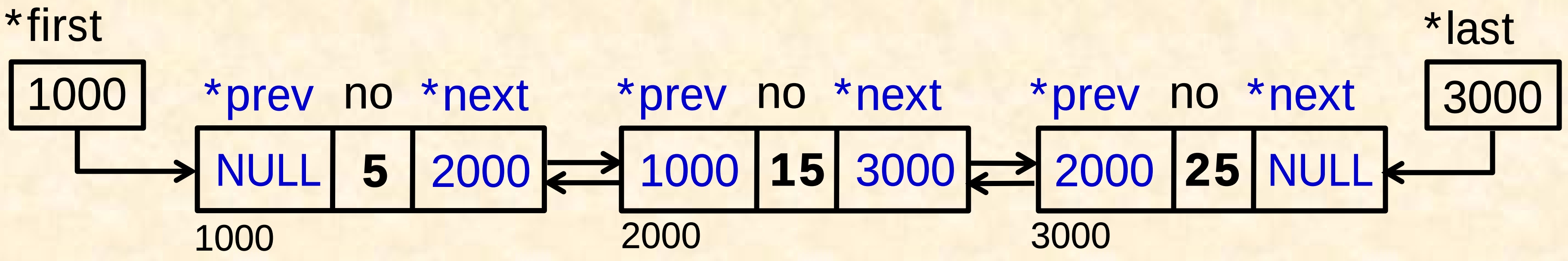
    - two, to point to its **previous** node."
  - Each node has **3** fields:

    Info + Next Pointer + Previous Pointer



DLL: Representation

- **Representation:**



- **Structure:**

```
struct dll
{
	struct  dll  *prev;
	int  no;
	struct  dll  *next;
};
```

DLL: Operations

- **Insert**
  - Insert at End (Append)
  - Insert at Front
  - Insert After
  - Insert Before
- **Traverse**
  - Forward
  - ~~Backward~~
- **Delete**
- **Search**
- **Count**

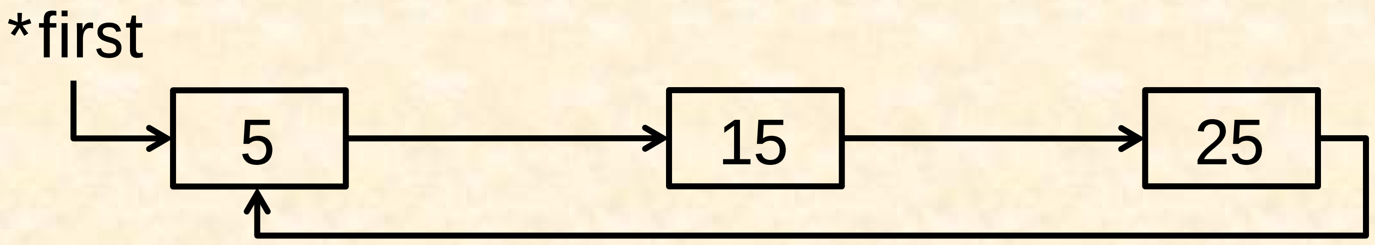
DLL: Advantage, Disad

- **Advantages:**
  - Traversal is possible in any direction – forward as well as backward
  - Any node can be visited from a given node
- **Disadvantages:**
  - Comparatively more complex
  - Each node requires more memory compared to SLL due to extra pointer

Circular Linked List

- **Def:**
  - "A linked list in which last node contains pointer –

    - back to the **first** node in a list."
- **Representation:**



Circular Linked List

- **Advantages:**
  - Any node can be visited from a given node
  - Not required to maintain address of first node
- **Disadvantages:**
  - Chance of infinite loop in processing list
  - Backward traversing is not possible in circular singly linked list

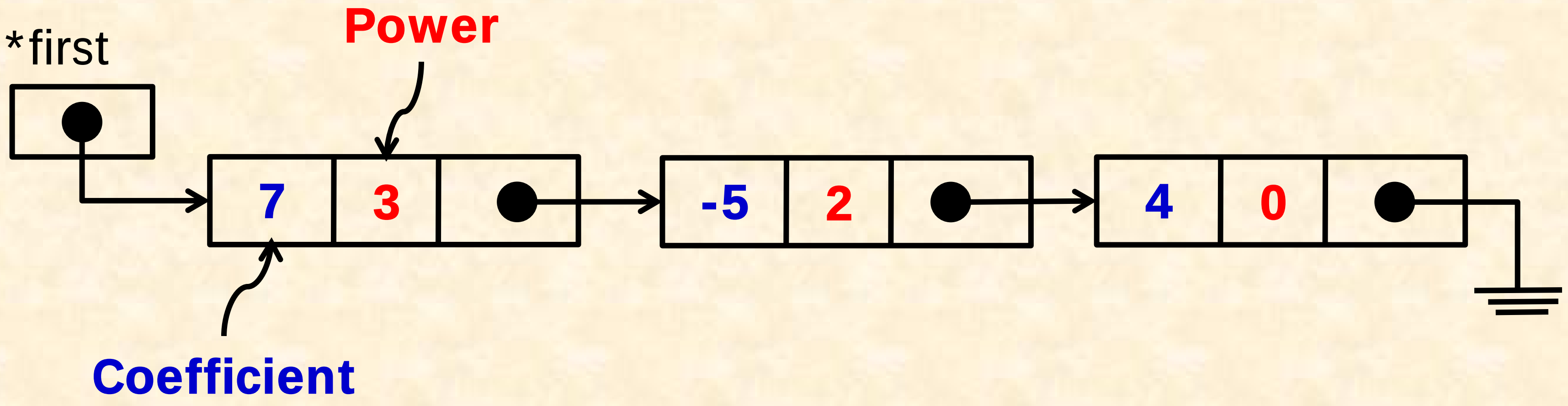
Applications of Linked List

- To manage data when total number of elements are unknown in advance
- To manage data when frequent insert-delete operations are possible
- To represent polynomial equations (provide example)

Polynomial Representation

Polynomial Equation: $7x^3 - 5x^2 + 4$

$= 7x^3 - 5x^2 + 4x^0$



Thank you…