CHARUSAT
CHAROTAR UNIVERSITY OF SCIENCE AND TECHNOLOGY

Accredited with Grade A by NAAC
Accredited with Grade A by KCG

cspit
Chandubhai S Patel Institute of Technology

CE261 : Data Structure and Algorithms

Topic: Introduction of Data Structures and Algorithms

Topic: Linked List

Prepared By,
Dr. Nikita Bhatt

Gate 2008

Suppose cursor refers to a node in a linked list. What statement changes cursor so that it refers to the next node?

(A)cursor++;
(B)cursor = link;
(C)cursor += link;
(D)cursor = cursor.link;

Answer : (D)

Gate 2009

What does the following function do for a given Linked List with first node as *head*?

```
void fun1(struct node* head)
    {
     if(head == NULL)
       return;
        fun1(head->next);
     printf("%d  ", head->data);
    }
```

(A) Prints all nodes of linked lists

(B) Prints all nodes of linked list in reverse order

(C) Prints alternate nodes of Linked List

(D) Prints alternate nodes in reverse order

Answer : (B)

Gate 2010

The following C function takes a singly-linked list as input argument. It modifies the list by moving the last element to the front of the list and returns the modified list. Some part of the code is left blank. Choose the correct alternative to replace the blank line.

```
typedef struct node {
  int value;
  struct node *next;}Node;
  Node *move_to_front(Node *head)
{
  Node *p, *q;
  if ((head == NULL|| (head->next == NULL))
    return head;
  q = NULL; p = head;
  while (p-> next !=NULL)
  {
    q = p;
    p = p->next;
  }
  _____
  return head;
}
```

Gate 2010 (Continue)

(A) q = NULL; p->next = head; head = p;

(B) q->next = NULL; head = p; p->next = head;

(C) head = p; p->next = q; q->next = NULL;

(D) q->next = NULL; p->next = head; head = p;

Answer : (D)

Gate 2008

The following C function takes a single-linked list of integers as a parameter and rearranges the elements of the list. The function is called with the list containing the integers 1, 2, 3, 4, 5, 6, 7 in the given order. What will be the contents of the list after the function completes execution?

```
struct node {   int value;  struct node *next; };
void rearrange(struct node *list) {
  struct node *p, * q;
  int temp;
  if ((!list) || !list->next)
          return;
  p = list;
  q = list->next;
           while(q)
          {
                    temp = p->value;
                   p->value = q->value;
                  q->value = temp;
                  p = q->next;
                  q = p?p->next:0;
           }
}
```

Gate 2008 (Continue)

(A) 1,2,3,4,5,6,7

(B) 2,1,4,3,6,5,7

(C) 1,3,2,5,4,7,6

(D) 2,3,4,5,6,7,1

Answer : (B)

Self Assessment-I

The following function reverse() is supposed to reverse a singly linked list. There is one line missing at the end of the function.

What should be added in place of “/*ADD A STATEMENT HERE*/”, so that the function correctly reverses a linked list.

```
/* Link list node */
struct node {     int data;     struct node* next; };
static void reverse(struct node** head_ref)
{
    struct node* prev   = NULL;
    struct node* current = *head_ref;
    struct node* next;
    while (current != NULL)
    {
        next  = current->next;
        current->next = prev;
        prev = current;
        current = next;
    }
    /*ADD A STATEMENT HERE*/
}
```

Self Assessment-I (Continue)

(A) *head_ref = prev;

(B) *head_ref = current;

(C) *head_ref = next;

(D) *head_ref = NULL;

Answer : (A)

Self Assessment-II

What is the output of following function for start pointing to first node of following linked list? 1->2->3->4->5->6

```
void fun(struct node* start)
{
  if(start == NULL)
    return;
  printf("%d  ", start->data);

  if(start->next != NULL )
    fun(start->next->next);
  printf("%d  ", start->data);
}
```

Self Assessment-II (continue)

(A) 1 4 6 6 4 1

(B) 1 3 5 1 3 5

(C) 1 2 3 5

(D) 1 3 5 5 3 1

Answer: (D)

Self Assessment-VI

Given code deletes the node which doesn't follow the ascending order sequence from this linked list : 3>>>4>>>5>>>6>>>7>>>9>>>8>>>10. Select correct statements for a1,a2,a3,and a4. The node structure is defined using this code:

Self Assessment-VI (continue)

```
struct node
{
int data;
struct node * link;
} *head, *p;
struct node *C= head;
while(C->data <= a1)
{
p=a2;
C=a3;
} a4;
```

Self Assessment-VI (continue)

(a)a1=C->data->link, a2=C, a3= C->link, a4=p->link=C->link

(b)a1=C->link->data, a2=C, a3= C->link, a4=p->link=C->link

(c)a1=C->link->data, a2=C->link, a3= C, a4=p->link=C->link

(d)a1=C->link->data, a2=C, a3= C->link, a4=C->link=P->link

Self Assessment-X

Find the output of below code for the linked list : 10->11->12->13->14->15->16.

```
struct Node
{
    int data;
    struct Node* next;
};
```

Self Assessment-X (Continue)

```
void print(struct Node* head)
{
   int count=0;
      while(head!=NULL)
      {
         if(count%2!=0)
            printf(" %d ",head->info);
         count+=3;
         head = head->link;
      }
}
```

Self Assessment-X(Continue)

(a)10 12 14 16

(b)11 14

(c)11 13 15

(d)10 13 16

Self Assessment-XI

Consider the following function that takes reference to head of a Doubly Linked List as parameter. Assume that a node of doubly linked list has previous pointer as prev and next pointer as next. Assume that reference of head of following doubly linked list is passed to above function

1 2 3 4 5 6.

Self Assessment-XI(continue)

```
void fun(struct node **head_ref){
    struct node *temp = NULL;
    struct node *current = *head_ref;
     while (current !=  NULL)    {
        temp = current->prev;
        current->prev = current->next;
        current->next = temp;
        current = current->prev;}
     if(temp != NULL )
        *head_ref = temp->prev;
}
```

Self Assessment-XI(continue)

What should be the modified linked list after the function call?

(A) 2 1 4 3 6 5

(B) 5 4 3 2 1 6.

(C) 6 5 4 3 2 1.

(D) 6 5 4 3 1 2

Answer : (C)

GATE-CS-2017

Consider the C code fragment given below.

```
typedef struct node {
        int data;
        node* next ; } node;
void join(node* m, node* n)
{
    node* p = n;
    while (p->next != NULL)
    {
        p = p->next;
    }
    p–>next = m;
}
```

GATE-CS-2017 (continue)

Assuming that m and n point to valid NULL- terminated linked lists, invocation of join will

(A) append list m to the end of list n for all inputs

(B) either cause a null pointer dereference or append list m to the end of list n

(C) cause a null pointer dereference for all inputs.

(D) append list n to the end of list m for all inputs.

Answer: (B)

Gate 2002

In the worst case, the number of comparisons needed to search a singly linked list of length n for a given element is

(A) log2 n

(B) n/2

(C) log2n – 1

(D) n

Answer: (D)

Self Assessment-III

Which of the following points is/are true about Linked List data structure when it is compared with array

(A) Arrays have better cache locality that can make them better in terms of performance.

(B) It is easy to insert and delete elements in Linked List

(C) Random access is not allowed in a typical implementation of Linked Lists

(D) The size of array has to be pre-decided, linked lists can change their size any time.

(E) All of the above

Gate 2018

Consider a singly linked list of the form where F is a pointer to the first element in the linked list and L is the pointer to the last element in the list. The time of which of the following operations depends on the length of the list?

(A) Delete the last element of the list

(B) Delete the first element of the list

(C) Add an element after the last element of the list

(D) Interchange the first two elements of the list

Answer: (A)

Self Assessment-IV

What are the time complexities of finding 8th element from beginning and 8th element from end in a singly linked list? Let n be the number of nodes in linked list, you may assume that n > 8.

(A) O(1) and O(n)

(B) O(1) and O(1)

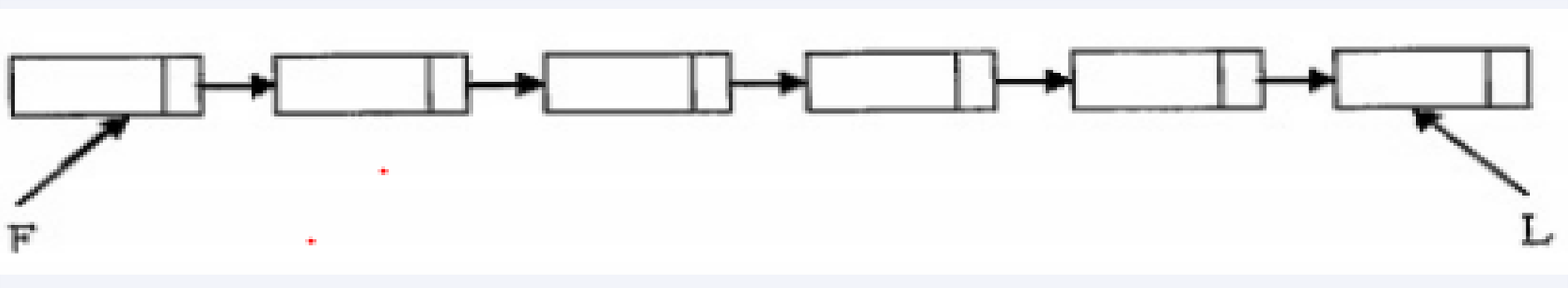
(C) O(n) and O(1)

(D) O(n) and O(n)

Answer: (A)

ISRO CS 2014

Consider a single linked list where F and L are pointers to the first and last elements respectively of the linked list. The time for performing which of the given operations depends on the length of the linked list?



(A) Delete the first element of the list

(B) Interchange the first two elements of the list

(C) Delete the last element of the list

(D) Add an element at the end of the list

Answer: (C)

Self Assessment-VII

How many comparisons are required to count the number of nodes in the linked list?

(a)1

(b)n

(c)logn

(d)n2

Self Assessment-XI

Consider an implementation of unsorted doubly linked list. Suppose it has its representation with a head pointer and tail pointer. Given the representation, which of the following operation can be implemented in O(1) time?

i) Insertion at the front of the linked list

ii) Insertion at the end of the linked list

iii) Deletion of the front node of the linked list

iv) Deletion of the end node of the linked list

a) I and II b) I and III c) I,II and III d) I,II,III and IV

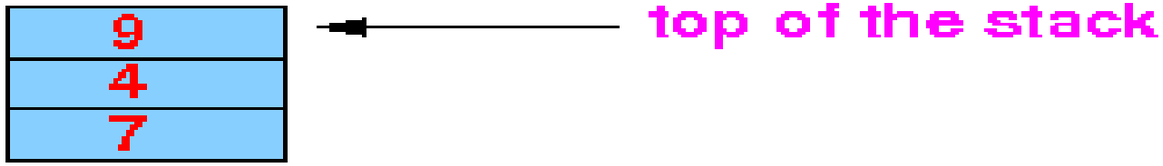
Answer: (d)

Applications of Linked List

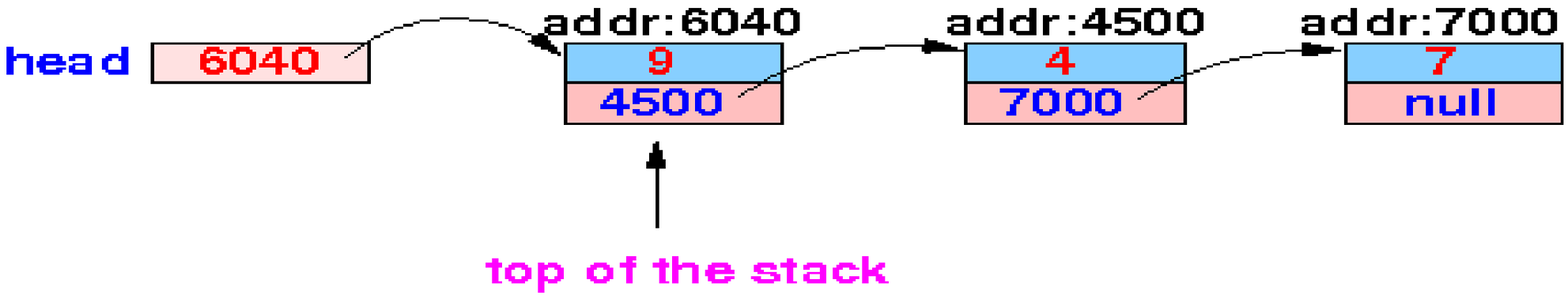
- **Implementation of stack data structure.**

Representing a stack with a list:

Stack:

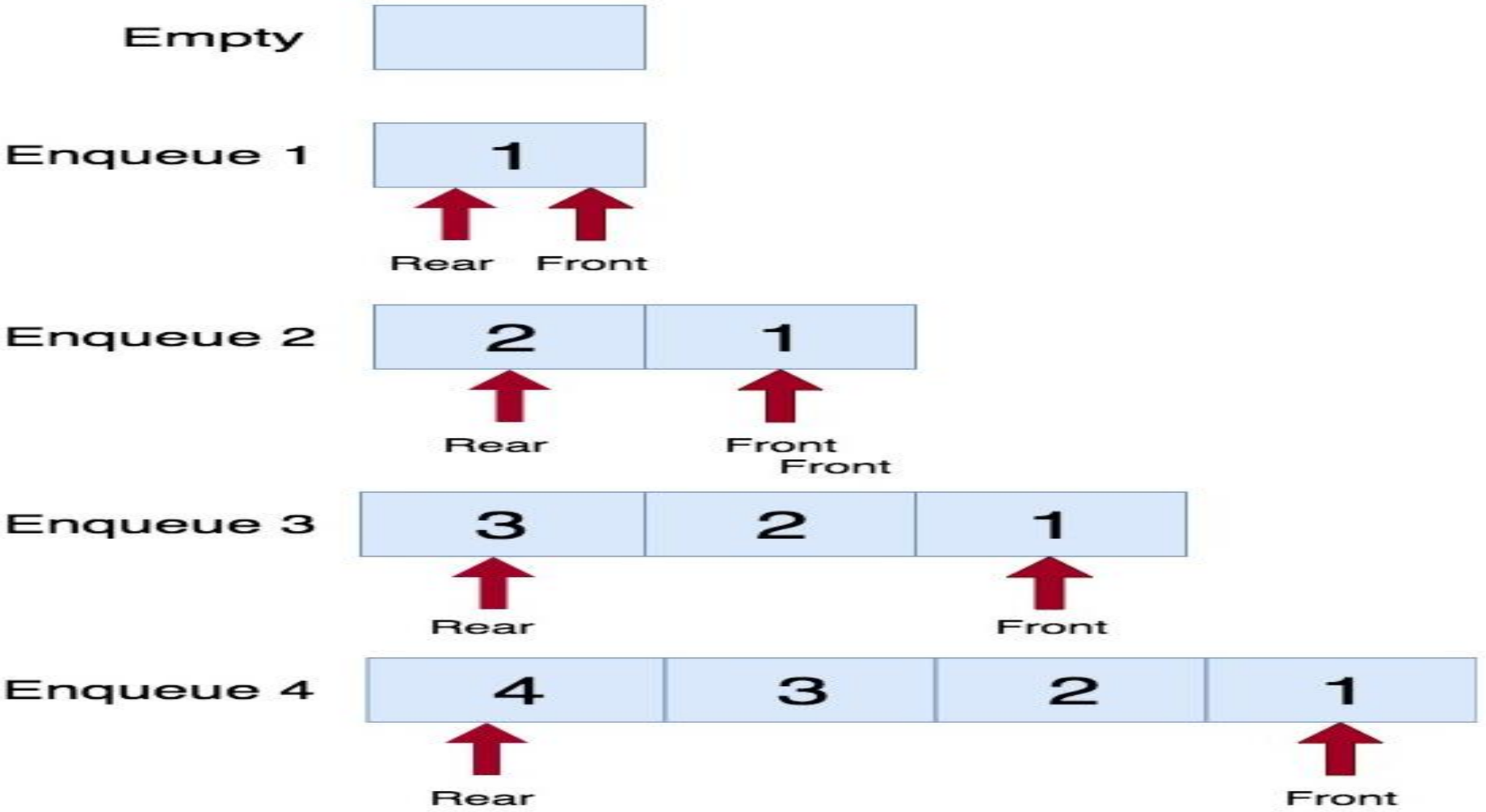


List:



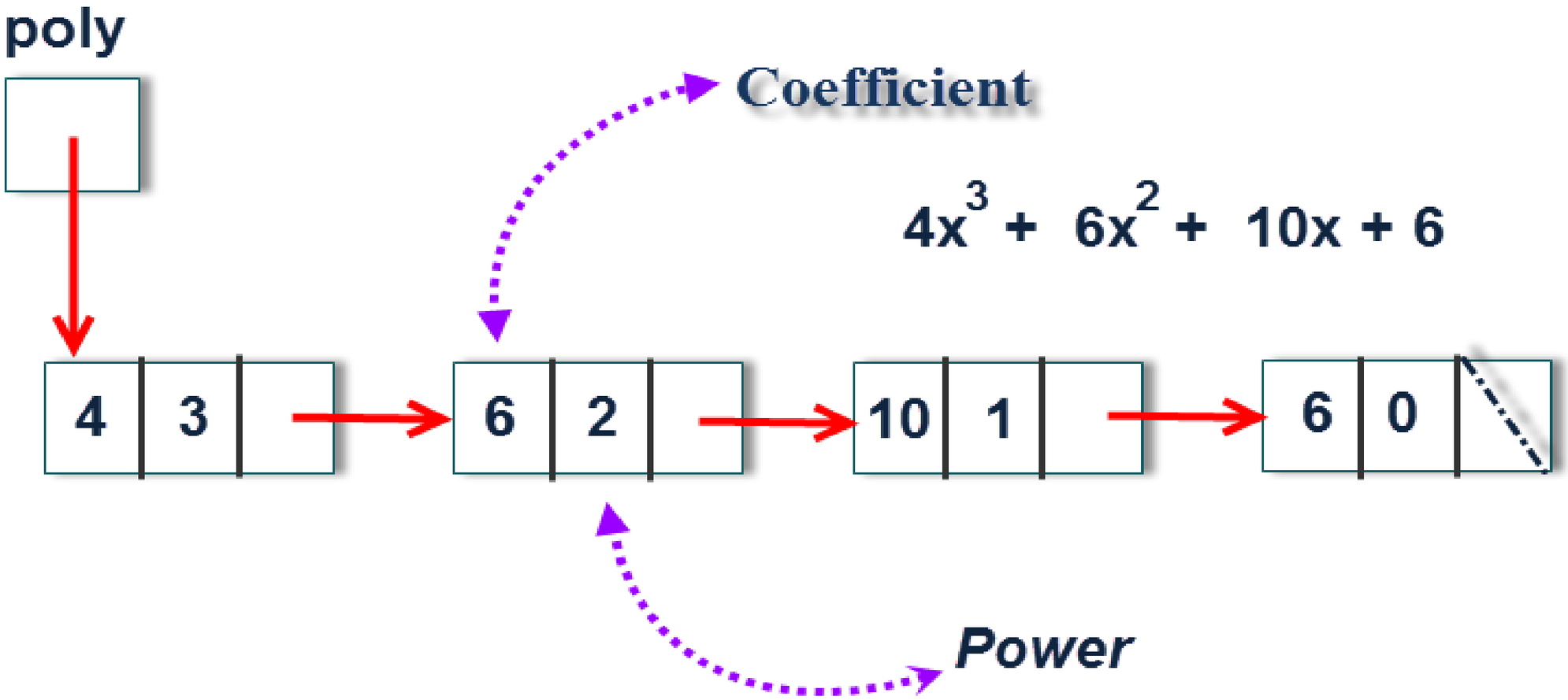
Applications of Linked List(continue)

- **Implementation of queue data structure.**



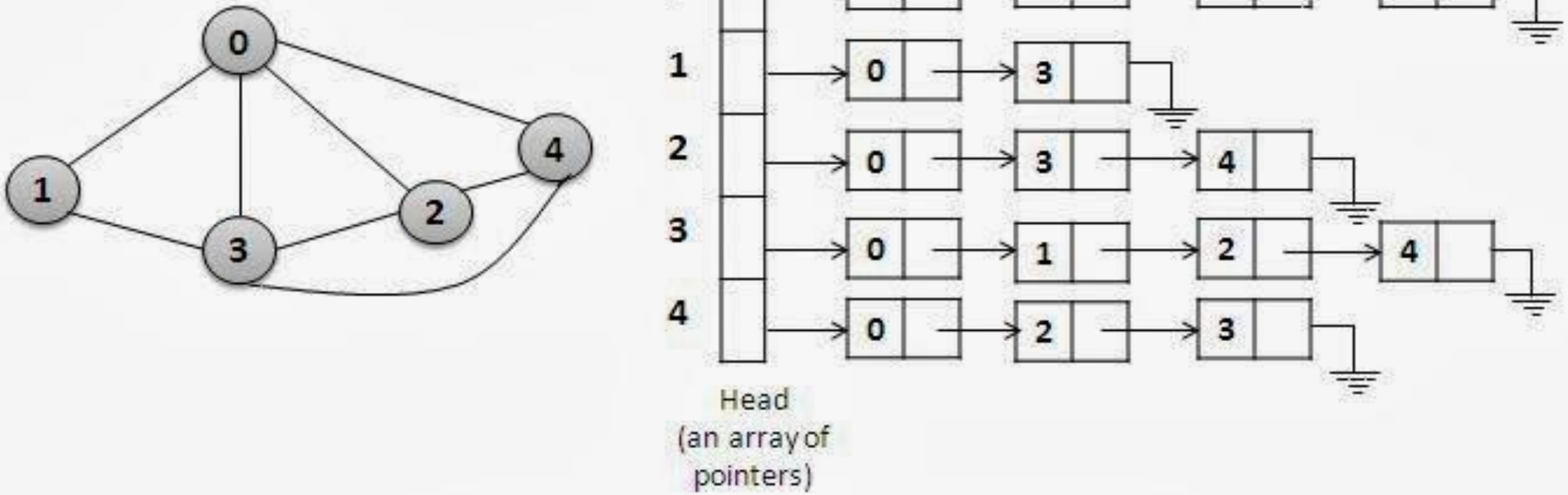
Applications of Linked List(continue)

- **Polynomial Expression Representation.**



Applications of Linked List(continue)

- **Graph can be represented in memory using linked list.**



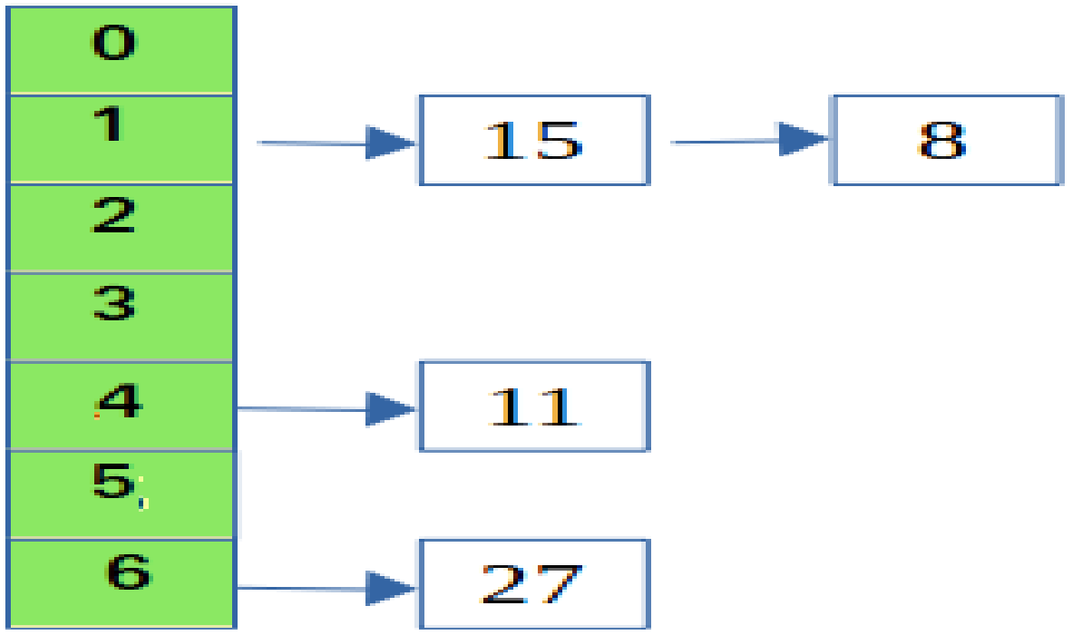
Adjacency List Representation of Graph

Applications of Linked List(continue)

- **Implementation of hash tables.**

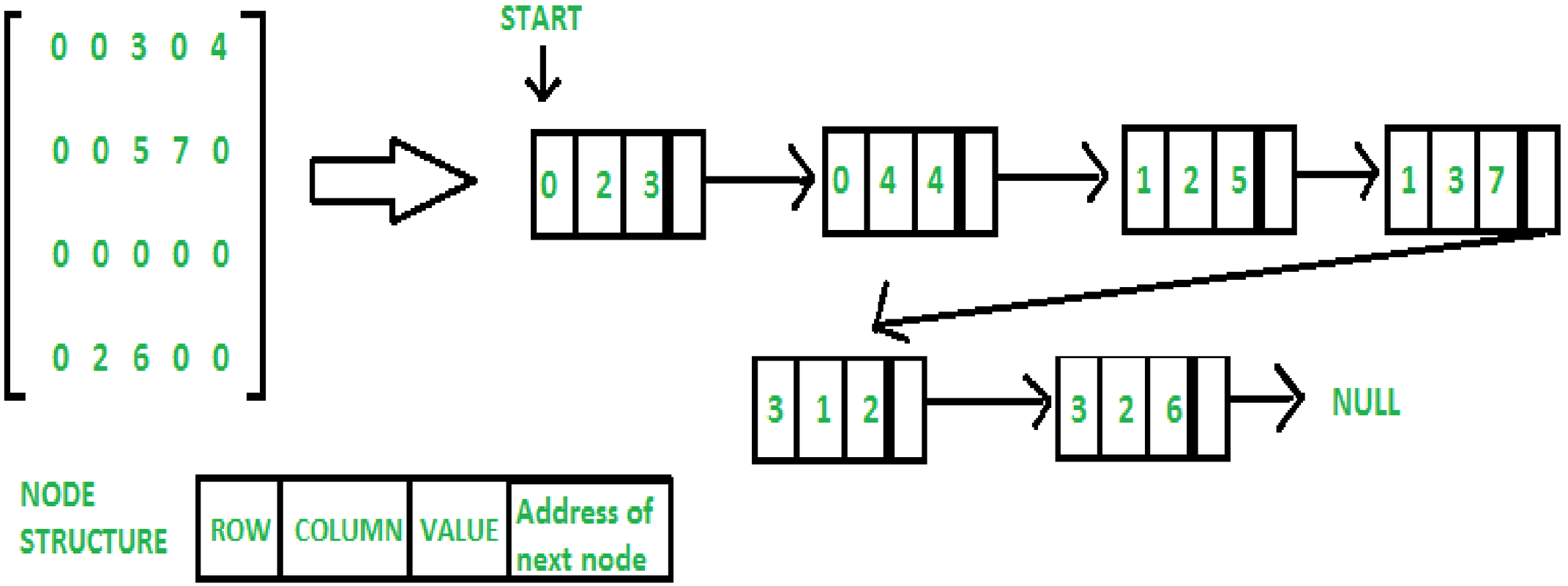
Let's say hash table with 7 buckets (0, 1, 2, 3, 4, 5, 6)

Keys arrive in the Order (15, 11 , 27 , 8)



Applications of Linked List(continue)

- **Implementation of sparse matrix.**

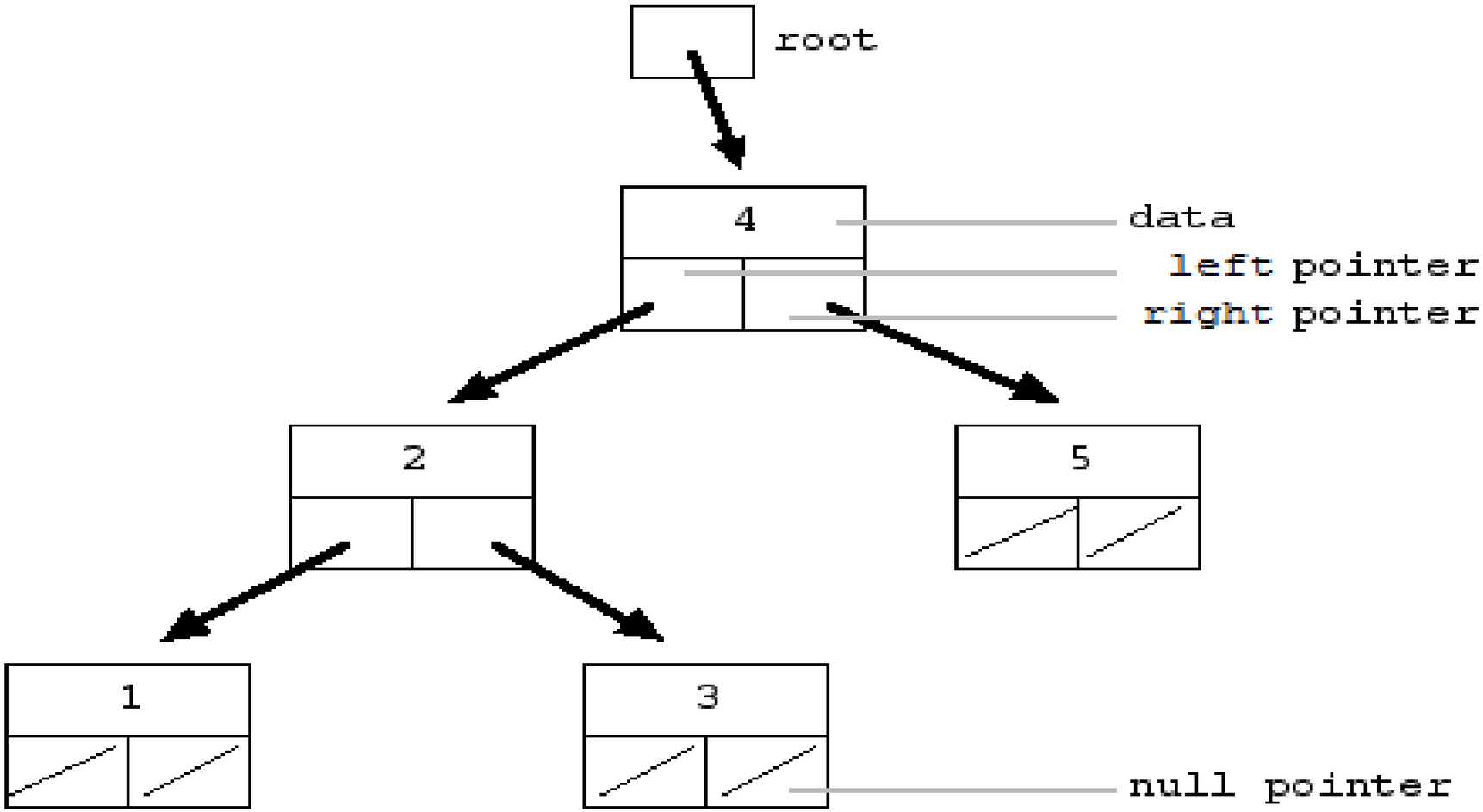


Applications of Linked List(continue)

- **Web browser**: The linked list is used by the browser to implement backward and forward navigation of visited web pages that is a back and forward button.
- **Navigation System:** The linked list is used in the navigation systems where front and back navigation is required.
- **Designing of MRU/LRU (Most/least recently used) cache:** The doubly linked List is also used to construct the cache.
- **Implementation of thread scheduler:** The doubly linked list is used to schedule the thread scheduler that chooses what process needs to run at which time.

Applications of Linked List(continue)

- **Implementation of tree data structure.**



Thank You