Name: Palash Mishra

Enrollno.: 201B172

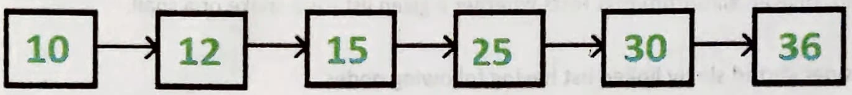
Due Sunday
07-11-21

JAYPEE UNIVERSITY OF ENGINEERING & TECHNOLOGY, GUNA
DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
Tutorial – 6 (Linked List)

Course: B. Tech Semester: III

Course Code &Name: **18B11CI311 – Data Structures**

1. **Consider an implementation of unsorted doubly linked list. Suppose it has its representation with a head pointer only. 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
2. Write an algorithm to insert an element at the second position in the linked list?
3. A variant of the linked list in which none of the node contains NULL pointer is?
4. Consider the following linked list



And following linked list representation

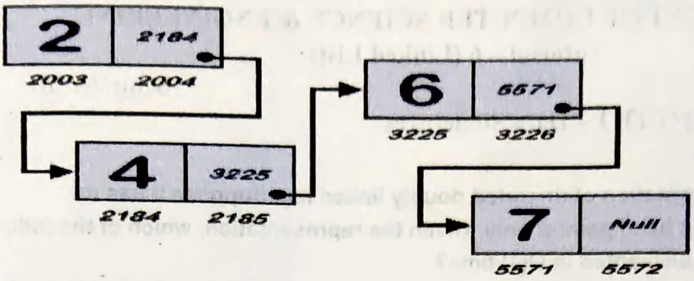
```
struct node {
        int data;
        struct node *next;
}*start ;
```

What will be printed by following statement?(Assume start is pointing to first node).

Printf("%d", start->next->next->next->data);

5. In doubly linked list each node has three fields, two fields for storing addresses and one for storing information of type **long double**. What will be the structure of such a node?
6. Let assume one doubly linked list of 5 nodes of type defined in question 5 then how much memory will be required for this linked list. What will be the PREV field of first node and NEXT field of last node in this linked list?
7. Write an efficient algorithm to find out kth node from the end of the linked list.

8.



If start is pointing to the first node of the linked list then consider the following statements

Start=start->next

temp=start->next

Current=temp->next

What will be the value of address field of temp and data field of current? ....

9. We are given a pointer to the first element of a linked list L. There are two possibilities for L, it either ends (snake) or its last element points back to one of the earlier elements in the list (snail). Give an algorithm that tests whether a given list L is a snake or a snail.

10. **Consider sorted singly linked list having following nodes**

    10->30->50->70->NULL

    You are given pointer to node 50 and a new node having value 40. Can you insert node 40 correctly in the list maintaining the ascending order?

11. Write an algorithm to delete a node from given location in doubly linked list.

Tutorial-6

(1.) (i.) and (iii)

i.e Insertion at the front of the linked list
Deletion of the front node of the linked list

(2.) Algorithm Ins_loc (START, Info, loc) // Input loc = 2

```
Node * Temp = START, * New_node
newNode = Allocate memory
newNode → data = Info
if (loc==1)
    newNode → Next = START
    START = newNode
else if( loc>=2 && Temp!= NULL)
    for (i=1; i<=loc-2; i++)
        Temp = Temp → Next
        If (Temp== NULL)
            Print "loc is greater"
            Return
    newNode → Next= Temp → Next
    Temp → Next = newNode
else
    Print "Invalid location"
```

(3) A variant of the linked list in which none of the node contains NULL pointer is Circular linked list.

(4) 25 will be printed on the screen

(5) Structure of doubly linked list :-

```
struct node {
        Struct node * PREV;
        Struct node * NEXT;
        long double info;
    };
```

(6) Considering machine using a 64-bit processor

So, each pointer here occupies 8 bytes of memory and infoof type long double occupies 16 bytes of memory.

∴ each node holds a memory of:

$2 \times 8 + 16 = 32$ bytes

and we have given 5 nodes Thus total memory in our list is 32×5 ie 160bytes.

PREV field of the first node and NEXT field of the last node have the NULL.

(7) Algorithm:

```
let given list be L
i=0
while L is not null
    if i==n
        break
    else
        move L to next node
        i+=1
```

```
Now, create a temp = Head of linked list
while L is not NULL
        move L to next node
        move Temp to next node
Now node pointing to temp is nth node from end.
```

Time Complexity : $O(n)$

8. Address field of Temp = 5571
Data field of current = 7

9. The goal here is to identify whether the given linked list has a cycle or not. If it does then it is a snail else it is a snake.

To achieve this we would implement 2 pointer algorithm. In this algorithm we start with 2 pointers at head of linked list. We move one pointer along the linked list with one element at a time whereas the outer pointer moves 2 elements at one time. If both the pointers ever meet then the list has a cycle.

Algorithm:

```
SnakeOrSnail(list)
{
   slow = list, fast = list      // pointers pointing the head
   while (slow && fast && fast→next)
   {
    slow= slow→next;      // moveslow One element to right
    fast= fast→next→next; // movefast ,,,,
    if (slow==fast) {     // If pointers ever meet
     return snail;        // cycle, hence a snail
    }
   };
   return Snake;          // nocycle, snake.
  }
```

(10.) No; we can't insert 40 inthe list mainlinging the ascending Order because we cannot iterate singly list backwards.

(11.)
```
Algo delete(struct DLLNode**head, loc)
    DLLNode *temp2, *temp=*head
    K=1
    if *head==NULL
      Print "Empty list"
      Return
    if loc==1
      *head=*head→next
    if *head!=NULL
      *head→PRev=NULL
      Delete(temp)
      Return
    while (K<loc && temp→next!=NULL)
      temp=temp→next
      k+=1
```

```
if k< loc-1
   Print "loc invalid"
   return

temp2 = temp → PRev
temp2→next = temp→next
if (temp→next)
   temp→next→PRev = temp2.
  Delete (temp)
  return.
```