**Assignment 4 | 7th January 2021**

Name: Rajarshi Lodh

Email: [rajarshilodh@gmail.com](mailto:rajarshilodh@gmail.com)

**Question 1**

Write a function “insert\_any()” for inserting a node at any given position of the linked list. Assume position starts at 0.

Function code:

struct node

{

int data;

node\* next;

};

node\* insert\_any(node\* head, int data, int pos)

{

int i = 0;

node \*curr, \*new;

new = (node\*)malloc(sizeof(node));

if(pos == 0)

{

new->data = data;

new->next = head->next;

head = new;

return head;

}

if(head == NULL)

{

new ->data = data;

return new;

}

else

{

do

{

curr = head;

curr = curr->next;

head = head->next;

i++;

}

while (i != pos-1);

new->next = curr->next;

new->data = data;

curr->next = new;

return head;

}

}

**Question 2**

Write a function “delete\_beg()” for deleting a node from the beginning of the linked list.

Function code:

struct node

{

    int data;

    struct node \*next;

};

int delete\_beg(struct node \*\*head)

{

    struct node \*tmp;

    int tmp\_val;

    if (\*head == NULL)

    {

        printf("Empty List\n");

        return -1;

    }

    tmp = \*head;

    tmp\_val = tmp->data;

    \*head = (\*head)->next;

    return (tmp\_val);

}

**Question 3**

Write a function “delete\_end()” for deleting a node from the end of the linked list.

Function code:

struct node

{

    int data;

    struct node \*next;

};

int delete\_end(struct node \*\*head)

{

    struct node \*tmp, \*prev;

    int val;

    if (\*head == NULL)

    {

        printf("Empty List\n");

        return -1;

    }

    tmp = \*head;

    while (tmp->next != NULL)

    {

        prev = tmp;

        tmp = tmp->next;

    }

    prev->next = NULL;

    val = tmp->data;

    return (val);

}

**Question 4**

In the Binary Search algorithm, it is suggested to calculate the mid as beg + (end - beg) / 2 instead of (beg + end) / 2. Why is it so?

**Answer:** Maximum time these both works same but there is problem with this approach of (beg + end) / 2, what if value of start or end or both is **INT\_MAX**, it will cause integer overflow. So beg + (end - beg) / 2 is generally suggested.

**Question 5**

Write the algorithm/function for Ternary Search.

**Function:**

int ternarySearch(int l, int r, int key, int ar[])

{

    if (r >= l) {

int mid1 = l + (r - l) / 3;

        int mid2 = r - (r - l) / 3;

if (ar[mid1] == key) {

            return mid1;

        }

        if (ar[mid2] == key) {

            return mid2;

        }

if (key < ar[mid1]) {

return ternarySearch(l, mid1 - 1, key, ar);

        }

        else if (key > ar[mid2]) {

  return ternarySearch(mid2 + 1, r, key, ar);

        }

        else {

return ternarySearch(mid1 + 1, mid2 - 1, key, ar);

        }

    }

  return -1;

}