JOEL NINAN JOHNSON
1BM19CS199
D2 - BATCH

Jninan...

classmate
Date
Page

23/11/2020

LAB-6 : SINGLY LINKED LISTS IMPLEMENTATION

Q. WAP to Implement Singly Linked List with following operations
a) Create a linked list
b) Insertion of a node at first position, at any position and at end of the list.
c) Deletion of a node at first position, at any position and at end of the list.
d) Display the contents.

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

push_front (struct Node** head, int new_data)
{
    struct Node *new_node = (struct Node*) malloc(sizeof(struct Node));
    new_node -> data = new_data;
    new_node -> next = *head;
    *head = new_node;
}

push_end (struct Node **head, int new_data)
{
    struct Node *new_node = (struct Node*) malloc(sizeof(struct Node));
    struct Node *last = *head;
    new_node -> data = new_data;
    new_node -> next = NULL;
    if (*head == NULL)
    {
        *head = new_node;
        return;
    }
```

JOEL NINAN JOHNSON
1BM19CS199
Finary..

23/11/2020

```
while (last -> next != NULL)
        last = last -> next
last -> next = new_node
return;
}

push_specific_pos (int data, int position)
{
        new_node
Struct Node * = (Struct Node*) malloc (sizeof (struct Node));
new_node -> data = data;
int i;
Struct Node * temp = head
if (position == 1)
{
    new_node -> next = temp
            head = new_node
            return
    }
    for (i=1; i < position-1; i++)
    {
        temp = temp -> next
    }
    new_node -> next = temp -> next
    temp -> next = new_node
}

print_LinkedList (Struct Node *node)
{
    while (node != NULL)
    {
        printf("%d", node -> data)
        node = node -> next
    }
}
```

JOEL NINAN JOHNSON
1BM19CS199

:- 23/11/2020

```
delete_front()
{
    struct Node *ptr;
    if (head == NULL)
    {
        printf("a List is Empty")
    }
    else
    {
        ptr = head
        head = ptr -> next
        free(ptr)
    }
}
ddelete_end()
{
    Struct Node *ptr, *pto1;
    if (head == NULL)
        List is Empty!!
    else if (head -> next == NULL)
    {
        head = NULL
        free (head)
    }
    else
    {
        ptr = head
        while (ptr -> next != NULL)
        {
            ptr1 = ptr
            ptr = ptr -> next
        }
```

JEL NINAN JOHNSON
IBM19CS199

23/11/2020

```
        ptr1 -> next = NULL
        free (ptr)
    }
}

delete_specific_pos()
{
    struct Node *ptr, *ptr1
    int i, position
    scanf("%d", &position)
    ptr = head
    for (i=0; i<position; i++)
    {
        ptr1 = ptr
        ptr = ptr -> next
        if (ptr == NULL)
        {
            "Less than required element in the list"
            return;
        }
    }
    ptr1 -> next = ptr -> next
    free(ptr)
}
```