5/02/2024

WEEK - 07

1) WAP to implement doubly linked list with following operations:- a) create b) insert a new node to the left of node
c) Delete specific node.

P:

```
struct node
{
    struct node *prev;
    int data;
    struct node *next;
};
struct node *start = NULL;
```

```
struct node *create_ll(struct node *start)
{
    struct node *new_node, *ptr;
    int num;
    printf("\nEnter -1 to end: ");
    printf("\nEnter data: ");
    scanf("%d", &num);
    while (num != -1)
    {
        if (start == NULL)
        {
            new_node = (struct node *)malloc(sizeof(struct node));
            new_node->prev = NULL;
            new_node->data = num;
            new_node->next = NULL;
            start = new_node;
        }
        else
        {
            ptr = start;
            new_node = (struct node *)malloc(sizeof(struct node));
            new_node->data = num;
            while(ptr->next != NULL)
                ptr = ptr->next;
            ptr->next = new_node;
            new_node->prev = ptr;
            new_node->next = NULL;
        }
        printf("\nEnter data: ");
        scanf("%d", &num);
    }
    return start;
```

```
struct node *display(struct node *start)
{
    struct node *ptr;
    ptr = start;
    while(ptr != NULL)
    { printf("\t %d", ptr->data);
      ptr = ptr->next;
    }
    return start;
}
struct node *insert_beg(struct node *start)
{
    struct node *new_node;
    int n;
    printf("\nEnter data: ");
    scanf("%d", &n);
    new_node = (struct node *)malloc(sizeof(struct node));
    new_node->data = num;
    start->prev = new_node;
    new_node->next = start;
    new_node->prev = NULL;
    start = new_node;
    return start;
}
struct node *insert_before(struct node *start)
{
    struct node *new_node, *ptr;
    int num, val;
    printf("\n Enter the value before which the
            data has to be inserted: ");
    scanf("%d", &val);
    if(start->data != val)
    {
      printf("\nEnter data: ");
      scanf("%d", &num);
      new_node = (struct node *)malloc(size of(struct node
      new_node->data = num;
      ptr = start;
      while(ptr->data != val)
          ptr = ptr->next;
      new_node->next = ptr;
      new_node->prev = ptr->prev;
      ptr->prev->next = new_node;
      ptr->prev = new_node;
    }
    else
    {   start = insert_beg(start);
    }
    return start;
}
struct node *delete_selected(struct node *start)
{
    struct node *ptr;
    int val;
    ptr = start;
    printf("\nEnter the value to be deleted: ");
    scanf("%d", &val);
    while(ptr->data != val)
    {   ptr = ptr->next;
    }
```

```
if (ptr->data == val)
{ ptr->prev->next = ptr->next;
  ptr->next->prev = ptr->prev;
  free(ptr);
}
else {
  printf("Node with %d value doesn't exist", val);
}
return start;
}
```

Output

choice: 1) Create 2) Display, 3) Add before
4) Delete given 5) Exit

option: 1
Enter -1 to end:
enter data: 1
Enter data: 2
Enter data: 3
Enter data: 4
Enter data: 5
Enter data: -1
Doubly linked list created

choice: 1) Create 2) Display 3) Add before
4) Delete given 5) Exit

option: 2

1 2 3 4 5

choice: 1) create 2) Display 3) Add before
4) Delete given 5) Exit

Enter the value before which data has to be inserted: 4
Enter data: 6

choice: 1) Create 2) Display 3) Add before
4) Delete given 5) Exit

option: 2

6 1 2 3 4 5

Choice: 1) Create 2) Display 3) Add before
4) Delete given 5) Exit

option: 4
Enter value to be deleted: 4

Choice: 1) Create 2) Display 3) Add before
4) Delete given 5) Exit

Option: 2

6 1 2 3 5

choice: 1) Create 2) Display 3) Add before
4) Delete given 5) Exit

Option: 5

Rev
5/2/24