12/01/2024

|WEEK-5|

1) 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 list.
   c) Display the contents of linked list.

S/p:

```
struct node
{
        int data;
        struct node *next;
    };
struct node *start = NULL;
struct node *create_ll (struct node *);
struct node *display(struct node *);
struct node * insert_beg(struct node *);
struct node * insert_end(struct node *);
struct node* insert_before_pos(struct node*);
struct node *create_ll (struct node *start)
{   struct node *new_node, *ptr;
    int num;
    printf(" \nEnter -1 to end");
    printf("\nEnter the data: ");
    scanf (" %d", &num);
    while (num!=-1)
    {
        new_node=(struct node*)malloc
                        (sizeof(struct node));
        new_node~~->~~data=num;
```

```
        if (start==NULL)
        {
            new_node->next=NULL;
            start =new_node;
        }
        else
        {   ptr=start;
            while (ptr->next!=NULL)
                ptr=ptr->next;
            ptr->next=new_node;
            new_node->next=NULL;
        }
        printf ("\n Enter the 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 num;
    printf ("\n Enter the data: ");
    scanf(" %d", &num);
    new_node=(struct node*)malloc(sizeof(struct node));
```

```
new_node -> data = num;
new_node -> next = start;
start = new_node;
return start;
};
struct node *insert_end(struct node *start)
{
    struct node *new_node, *ptr;
    int num;
    printf("\n Enter the data: ");
    scanf("%d", &num);
    new_node = (struct node *) malloc(size of (struct node));
    new_node -> data = num;
    new_node -> next = NULL;
    ptr = start;
    if (start == NULL)
    {
        start = new_node;
    }
    else
    {
        while (ptr -> next != NULL)
                ptr = ptr -> next;
        ptr -> next = new_node;
        return start;
    }

};
struct node *insert_before(struct node *start)
{   struct node *new_node, *ptr, *preptr;
    int num, val;
    printf("\n enter the data: ");
    scanf("%d", &num);
    printf("\n enter the value before which data
                has to be inserted: ");
    scanf("%d", &val);
    new_node = (struct node *) malloc(sizeof(struct node));
    new_node -> data = num;
    ptr = start;
    while (ptr -> data != val)
    {   preptr = ptr;
        ptr = ptr -> next;
    }
    preptr -> next = new_node;
    new_node -> next = ptr;
    return start;
};
```

2) Q:- Deletion of first element, specified element & last element in the list & Display the contents.

I/p:-

```
struct node *display(struct node *);
struct node *delete_beg(struct node *);
struct node *delete_end(struct node *);
struct node *delete ~~before~~ after pos(struct node *);
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 *delete_beg(struct node *start)
{
    struct node *ptr;
    ptr=start;
    start=start->next;
    free(ptr);
    return start;
};
struct node *delete_end(struct node *start)
{
    struct node *ptr, *preptr;
    ptr=start;
    while(ptr->next!=NULL)
    {
        preptr=ptr;
        ptr=ptr->next;
    }
    preptr->next=NULL;
    free(ptr);
    return start;
}
struct node *delete_after_pos(struct node *start)
    struct node *ptr, *preptr;
    int val;
    printf("\n Enter the value after which the
                node has to be deleted: ");
    scanf("%d", &val);
    ptr=start;
    preptr=ptr;
while(preptr->data!=val)
{   preptr=ptr;
    ptr=ptr->next,
}
preptr->next=ptr->next,
free(ptr),
return start;
};
```

i) 1. create a linked list 2. display 3. insert_beginning
4. insert_end 5. insert_random

```
enter the choice: 1
enter -1 to end
enter the data: 1
enter the data: 2
enter the data: 3
enter the data: -1
Linked list created
enter the choice: 3
enter data: 4
enter the choice: 2
~~enter data:~~ 4 1 2 3
enter the choice: 4
enter the data: 5
enter the choice: 2

   4  1 2 3 5
enter the choice: 5
enter before which element new
            element is inserted: 3
enter data: 6
enter choice: 2

    4  1  2  6  3  5

2)
  1. create a list  2. display  3. del-beg
  4. del_end  5. delete_node
  enter choice: 1
  enter -1 to end
  enter data: 1
  enter data: 2
  enter data: 1
  enter data: -1
  Linked list created
  enter choice: 2

       1  2  1

  enter choice: 3
  enter choice: 2

       2 1
  enter choice: 4
  enter choice: 2

       2

  ~~enter ch~~
```

[illegible]
22/1/24