5/2/2024

1. WAP to implement doubly link list with primitive operations.

a) create a doubly linked list

b) Insert a new node to the left of the node.

c) Delete the node based on a specific value.

```
#include <stdio.h>
#include<stdlib.h>

struct node
{
    int data;
    struct node *next;
    struct node *prev;
};

struct node *head;

void createll()
{
    struct node *new_node,*ptr;
    int num;
    printf("Enter -1 to exit\n");
    while(num!=-1)
    {
        printf("Enter the num:");
        scanf("%d",&num);
        new_node=(struct node*)malloc(sizeof(struct node));
        new_node->data=num;
        if(head==NULL)
        {
            head=new_node;
            new_node->next=NULL;
            new_node->prev=NULL;
```

```
else
{
    ptr = head
    while (ptr->next != NULL)
    {
      ptr = ptr->next;
    }
    ptr->next = new_node;
    new_node->prev = ptr;
    new_node->next = NULL;
  }
}
}

void insert_left()
{
  struct node *new_node, *ptr;
  int val, num;
  new_node = (struct node *) malloc (sizeof (struct node));
  printf("enter a value to insert at left:");
  scanf("%d", &val);
  printf("Enter the value of node:");
  scanf("%d", &num);
  new_node->data = val;
  ptr = head;
  while (ptr->data != num)
  {
      ptr = ptr->next;
  }
  ptr->prev->next = new_node;
  new_node->prev = ptr->prev;
  new_node->next = ptr;
  ptr->prev = new_node;
}
```

```
void display()
{
    if (head == NULL)
        printf("the list empty");
    else
    {
        ptr = head;
        while (ptr->next != NULL)
        {
            printf("%d", ptr->data);
            ptr = ptr->next;
        }
    }
}

void del()
{
    struct node *ptr;
    int val;
    printf("enter the value to be deleted");
    scanf("%d", &val);
    ptr = head;
    if (head->data == val)
    {
        ptr = ptr->next;
        head = ptr;
    }
    else
    {
        while(ptr->data != val)
        {
            ptr = ptr->next;
        }
        ptr->prev->next = ptr->next;
        ptr->next->prev = ptr->prev;
    }
    free(ptr);
```

o/p:

1. create - 11
2. insert - left
3. delete
4. display
5. exit

1.

Enter -1 to exit:
Enter the num: 32
Enter the num: 46
Enter the num: 65
Enter the num: -1

4.

32 → 46 → 65

2

enter a value to insert at left: 99
enter the val of node: 46

4

32 → 99 → 46 → 65

3

enter the value to be deleted: 46

4

32 → 99 → 65

Lab
5/2/24

```c
#include <stdio.h>
#include <stdlib.h>

struct node
{
    int data;
    struct node *next;
    struct node *prev;
};

struct node *head;

void create_ll()
{
    struct node *new_node, *ptr;
    int num;
    printf("Enter -1 to exit.. \n");
    // printf("Enter the num: ");
    // scanf("%d", &num);
    while (num != -1)
    {
        printf("Enter the num: ");
        scanf("%d", &num);
        new_node = (struct node *)malloc(sizeof(struct node));
        new_node->data = num;
        if (head == NULL)
        {
            head = new_node;
            new_node->next = NULL;
            new_node->prev = NULL;
        }
        else
        {
            ptr = head;
            while (ptr->next != NULL)
            {
                ptr = ptr->next;
            }
            ptr->next = new_node;
            new_node->prev = ptr;
            new_node->next = NULL;
        }
    }
}
```

```c
void insert_left()
{
    struct node *new_node, *ptr;
    int val, num;
    new_node = (struct node *)malloc(sizeof(struct node));
    printf("enter a value to insert at left:");
    scanf("%d", &val);
    printf("Enter the value of node:");
    scanf("%d", &num);
    new_node->data = val;
    ptr = head;
    if (head == NULL)
    {
        printf("list is empty!");
    }
    else
    {
        while (ptr->data != num)
        {
            ptr = ptr->next;
        }
        ptr->prev->next = new_node;
        new_node->prev = ptr->prev;
        new_node->next = ptr;
        ptr->prev = new_node;
    }
}
```

```c
void display()
{
    struct node *ptr;
    if (head == NULL)
    {
        printf("Linked list is empty!");
    }
    else
    {
        ptr = head;
        while (ptr->next != NULL)
        {
            printf("%d->", ptr->data);
            ptr = ptr->next;
        }
    }
}

void del()
{
    struct node *ptr;
    int val;
    printf("enter the value to be deleted:");
    scanf("%d", &val);
    ptr = head;
    if (head->data == val)
    {
        ptr = ptr->next;
        head = ptr;
    }
    else
    {
        while (ptr->data != val)
        {
            ptr = ptr->next;
        }
        ptr->prev->next = ptr->next;
        ptr->next->prev = ptr->prev;
        free(ptr);
    }
}

int main()
```

```c
int main()
{
    int value, choice;
    while (1)
    {
        printf("---------------MENU---------------\n");
        printf("1.create_ll\n 2.insert_left\n 3.delete\n 4.display\n 5.exit\n");
        scanf("%d", &choice);
        switch (choice)
        {
        case 1:
            create_ll();
            break;

        case 2:
            insert_left();
            break;

        case 3:
            del();
            break;

        case 4:
            display();
            break;

        case 5:
            exit(0);
            break;

        default:
            printf("wrong input!\n");
            break;
        }
    }
}
```

```
---------------MENU----------------
1.create_ll
 2.insert_left
 3.delete
 4.display
 5.exit
1
Enter -1 to exit..
Enter the num: 32
Enter the num: 46
Enter the num: 65
Enter the num: -1
---------------MENU----------------
1.create_ll
 2.insert_left
 3.delete
 4.display
 5.exit
4
32->46->65->---------------MENU----------------
1.create_ll
 2.insert_left
 3.delete
 4.display
 5.exit
2
enter a value to insert at left:99
Enter the value of node:46
---------------MENU----------------
1.create_ll
 2.insert_left
 3.delete
 4.display
 5.exit
4
32->99->46->65->---------------MENU----------------
1.create_ll
 2.insert_left
 3.delete
 4.display
```

```
1.create_ll
 2.insert_left
 3.delete
 4.display
 5.exit
3
enter the value to be deleted:46
---------------MENU----------------
1.create_ll
 2.insert_left
 3.delete
 4.display
 5.exit
4
32->99->65->----------------MENU----------------
1.create_ll
 2.insert_left
 3.delete
 4.display
 5.exit
```