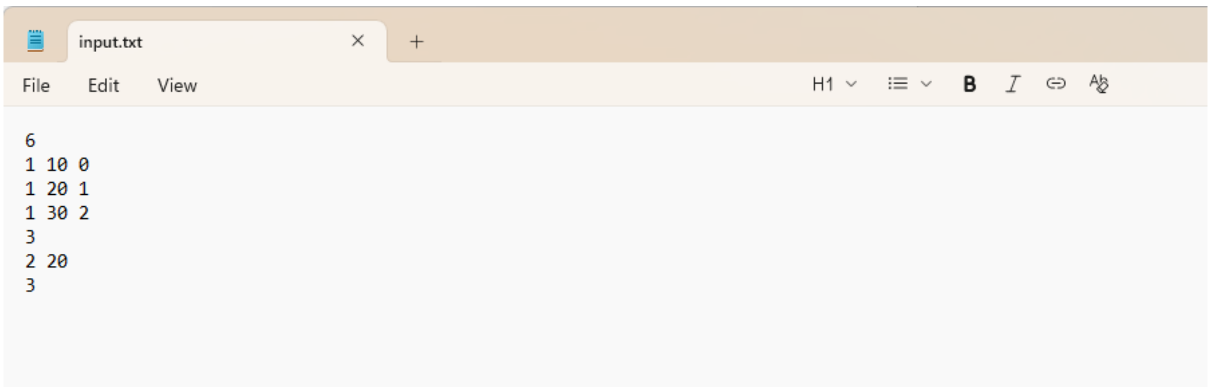
WAP to Implement doubly link list with primitive operations
a) Create a doubly linked list.
b) Insert a new node to the left of the node.
c) Delete the node based on a specific value
d) Display the contents of the list

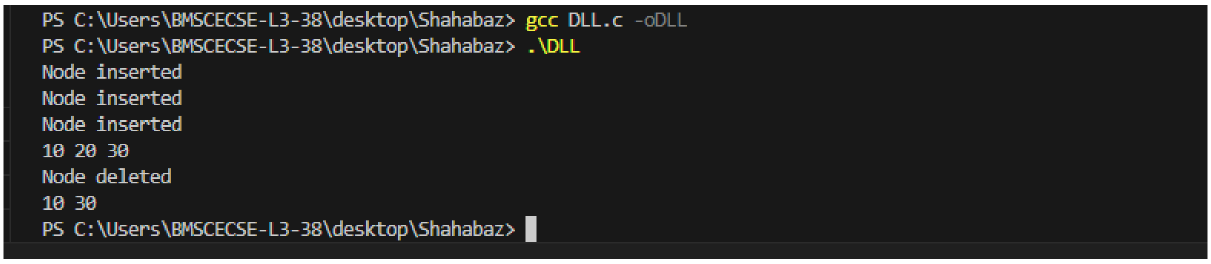
```c
#include<stdio.h>
#include<stdlib.h>
struct node{
    struct node *prev;
    int data;
    struct node *next;
};

struct node *head = NULL;
struct node *createNode(int data)
{
    struct node *nn = (struct node*)malloc(sizeof(struct node));
    if (nn == NULL){
        printf("No memory allocated\n");
        exit(0);
    }
    nn->prev = NULL;
    nn->next = NULL;
    nn->data = data;
    return nn;
}
void insert_specified(int item, int loc){
    struct node *ptr = createNode(item);
    if(loc == 0){
        ptr->next = head;
        if(head != NULL)
            head->prev = ptr;
        head = ptr;
        printf("Node inserted\n");
        return;
    }

    struct node *temp = head;
    for(int i = 0; i < loc - 1; i++){
        if(temp == NULL){
            printf("Can't insert\n");
            return;
        }
        temp = temp->next;
    }

    if(temp == NULL){
        printf("Can't insert\n");
        return;
    }
    ptr->prev = temp;

    ptr->next = temp->next;

    if(temp->next != NULL)
        temp->next->prev = ptr;

    temp->next = ptr;

    printf("Node inserted\n");
}
void delete_spec_val(int val){
    struct node *temp = head;

    if(head == NULL){
        printf("List empty\n");
        return;
    }
    while(temp != NULL && temp->data != val){
        temp = temp->next;
    }

    if(temp == NULL){
        printf("Value not found\n");
        return;
    }
    if(temp == head){
        head = temp->next;
        if(head != NULL)
            head->prev = NULL;
        free(temp);
        printf("Node deleted\n");
        return;
    }
    if(temp->prev != NULL)
        temp->prev->next = temp->next;

    if(temp->next != NULL)
        temp->next->prev = temp->prev;

    free(temp);

    printf("Node deleted\n");
}
int display(){
    struct node *ptr = head;

    if(head == NULL){
        printf("Empty list\n");
        return 0;
    }

    while(ptr != NULL){
        printf("%d ", ptr->data);
        ptr = ptr->next;
    }
    printf("\n");
    return 0;
}
int main() {
    FILE *fp = fopen("input.txt", "r");
    if(fp == NULL){
        printf("Could not open input file\n");
        return 0;
    }

    int t;
    fscanf(fp, "%d", &t);

    while(t--){
        int choice;
        fscanf(fp, "%d", &choice);

        switch(choice){
            case 1:{
                int val, loc;
                fscanf(fp, "%d %d", &val, &loc);
                insert_specified(val, loc);
                break;
            }

            case 2:{
                int val;
                fscanf(fp, "%d", &val);
                delete_spec_val(val);
                break;
            }

            case 3:
                display();
                break;

            default:
                printf("Invalid choice\n");
        }
    }
    fclose(fp);
    return 0;
}
```

INPUT FILE:-



```
6
1 10 0
1 20 1
1 30 2
3
2 20
3
```

OUTPUT:-



```
PS C:\Users\BMSCECSE-L3-38\desktop\Shahabaz> gcc DLL.c -oDLL
PS C:\Users\BMSCECSE-L3-38\desktop\Shahabaz> .\DLL
Node inserted
Node inserted
Node inserted
10 20 30
Node deleted
10 30
PS C:\Users\BMSCECSE-L3-38\desktop\Shahabaz>
```