Exp. 6.

```
        }
        ptr = ptr->next;
    }
    if(ptr == NULL){
        printf("Position not found.\n");
        free(temp);
        return;
    }
    temp->next = ptr->next;
    temp->prev = ptr;

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

    ptr->next = temp;


void delete_pos(){
    struct node *ptr;
    int pos, i;

    if(start == NULL){
        printf("\nList is empty\n");
        return;
    }

    printf("Enter position to delete: ");
    scanf("%d",&pos);

    if(pos == 1){
        ptr = start;
        start = start->next;
        if(start != NULL)
            start->prev = NULL;
        free(ptr);
        return;
    }

    ptr = start;
    for(i=0; i<pos-1; i++){
        if(ptr == NULL){
            printf("Position not found.\n");
            return;
        }
        ptr = ptr->next;
    }

    if(ptr == NULL){
        printf("Position not found.\n");
        return;
    }

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

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

    free(ptr);
}

void delete_all(){
    struct node *ptr;
    while(start != NULL){
        ptr = start;
        start = start->next;
        free(ptr);
    }
```

```
    printf("\nAll nodes deleted. List is now empty.\n");
}

void count_elements(){
    struct node *ptr = start;
    int count = 0;
    while(ptr != NULL){
        count++;
        ptr = ptr->next;
    }
    printf("Number of elements in the list: %d\n", count);
}

void reverse_list(){
    struct node *current = start, *temp = NULL;

    if(start == NULL){
        printf("List is empty.\n");
        return;
    }

    while(current != NULL){
        temp = current->prev;
        current->prev = current->next;
        current->next = temp;
        current = current->prev;
    }

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

    printf("List reversed successfully.\n");
```

```
1. create
2. display
3. insert at begin
4. insert at end
5. insert at any position
6. delete at position
7. delete all
8. count elements
9. reverse list
10. exit
Enter your choice: 1
Enter data: 12

1. create
2. display
3. insert at begin
4. insert at end
5. insert at any position
6. delete at position
7. delete all
8. count elements
9. reverse list
10. exit
Enter your choice: 3
Enter data: 23

1. create
2. display
3. insert at begin
4. insert at end
5. insert at any position
6. delete at position
7. delete all
8. count elements
```

```
//writefile all.c
#include<stdio.h>
#include<stdlib.h>
//doublylinkedlist
void create();
void display();
void insert_begin();
void insert_end();
void insert_pos();
void delete_pos();
void delete_all();
void count_elements();
void reverse_list();

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

struct node *start = NULL;

int main(){
    int choice;
    while(1){
        printf("\n1. create\n");
        printf("2. display\n");
        printf("3. insert at begin\n");
        printf("4. insert at end\n");
        printf("5. insert at any position\n");
        printf("6. delete at position\n");
        printf("7. delete all\n");
        printf("8. count elements\n");
        printf("9. reverse list\n");
```

```
9. reverse list
10. exit
Enter your choice: 2

List elements: 23 12

1. create
2. display
3. insert at begin
4. insert at end
5. insert at any position
6. delete at position
7. delete all
8. count elements
9. reverse list
10. exit
Enter your choice: 10
```