

start->prev = NULL;

free(ptr);

ptr = ptr->next;

}

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

temp->next = ptr;

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

```
} 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);  
}
```

```
old_delete_pos(){  
    struct node *ptr;  
    int pos, i;  
    if(start == NULL){  
        printf("\nList is empty\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;  
        if(start == NULL)  
            free(ptr);  
    }
```

}

Exp. 6.

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

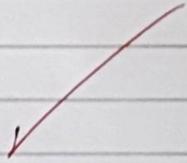
```
printf("\nAll nodes deleted. List is now empty.\n");  
  
void count_elements(){  
    struct node *ptr = start;  
    int count = 0;  
    while(ptr != NULL){  
        count++;  
        ptr = ptr->next;  
    }  
    printf("Number of elements in the list: %d\n", count);  
  
}  
  
void reverse_list(){  
    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->next;  
    else  
        printf("List reversed successfully.\n");  
}
```

```
#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("1. 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