6) Write prog Implement single linked list to simulate Stack & Queue Operations
W.A.P. to insert, delete in single linked list

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

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

void create() {
    int choice = 1;
struct node *newnode;
struct node *temp;
while (choice != 0) {
 newnode = (struct node *)malloc(sizeof(struct node));
 printf("Enter the data");
 scanf("%d", &newnode->data);
 newnode->next = 0;
 if (head == 0) {
        head = temp = newnode;
 }
else { temp->next = newnode;
       temp = newnode;
 }
printf("Do you want to continue");
scanf("%d", &choice);
 }
```

```
void insertatbeginning() {
    struct node *newnode;
    newnode = (struct node*) malloc(sizeof(struct node));
    printf("Enter the data to be inserted at front");
    scanf("%d", &newnode->data);
    newnode->next = head;
    head = newnode;
}
void insertatend() {
    struct node *newnode, *last;
    newnode = (struct node*) malloc(sizeof(struct node));
    printf("Enter the data to be inserted at end");
    scanf("%d", &newnode->data);
    newnode->next = 0;
    last = head;
    if (head == 0) {
        head = newnode;
    }
else {
        while (last->next != 0) {
            last = last->next;
        }
        last->next = newnode;
}
}
```

```
void insatbeginlocation() {
    struct node *newnode, *location;
    newnode = (struct node*) malloc (sizeof (struct
                                        node));
    printf ("Enter the data to be insert");
    scanf ("%d", &newnode->data);
    location = head;
    int i=1, pos;
    printf ("Enter the position when you want
                to insert");
    scanf ("%d", &pos);
    while (i<(pos-1)){
        location = location->next;
        i++;
    }
    newnode->next = location->next;
    location->next = newnode;
}
// deletion code
void deletatbegin() {
    struct node *temp;
    if (head==0) {
        printf ("list is empty");
    }
    else {
        temp = head;
        head = head->next;
    }
    free (temp);
}
```

```
void deletatend() {
struct node * temp, * previous;
if (head == NULL) {
    printf("list is empty");
}
else {
    temp = head;
    while (temp -> next != NULL) {
        previous = temp;
        temp = temp -> next;
    }
    if (temp == head) {
        head = NULL;
    }
    else {
        previous -> next = NULL;
    }
free(temp);
}
}
void deletatspecific() {
struct node * temp, * previous;
int pos;
int i=2;
printf("Enter the position to delete");
```

```
        scanf("%d", &pos);
        temp = head;
        while(i < pos) {
            previous = temp;
            temp = temp->next;
            i++;
        }
        if (temp == head) {
            head = 0;
        }
        else { previous->next = temp->next; }
        free(temp);
    }
    void display() {
        struct node *temp;
        temp = head
        while (temp->next != 0) {
            printf("%d", temp->data);
            temp = temp->next;
        }
    }
    void main() {
        create();
        display();
        insertatbeginning();
        display();
        insertatend();
        display();
```

```
        insertatgivenlocation();
        display();
        deleteatbegin();
        display();
        deleteatend();
        display();
        deleteatspecific();
        display();
    }
```

Output →

Enter the data 1

Do you want to continue 1

Enter the data 2

Do you want to continue 1

Enter the data 3

Do you want to continue 1

Enter the data 4

Do you want to continue 0

1 2 3 4

Enter the data to be inserted at the first

0 1 2 3 4

Enter the data to be inserted at end

0 1 2 3 4 5

Enter the data to be insert

6

Enter the position where you want to insert

3

0 1 6 2 3 4 5

1 6 2 3 4 5 // Delete at begin

1 6 2 3 4 // Delete at end

Enter the Position to delete 4
1624