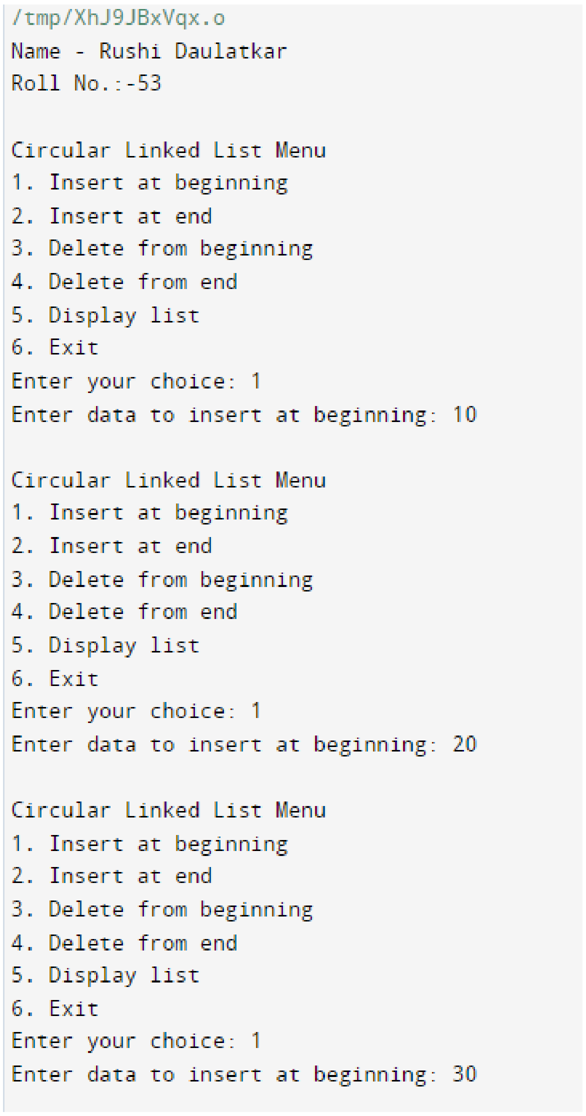
EXPERIMENT NO. 9

```
//The Code Is As Follows :-


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


                                                    // Define the structure of a node
struct Node {
    int data;
    struct Node *next;
};


                                                          // Function prototypes
void insertAtStart(struct Node **head_ref, int new_data);
void insertAtEnd(struct Node **head_ref, int new_data);
void deleteFromStart(struct Node **head_ref);
void deleteFromEnd(struct Node **head_ref);
void display(struct Node *head);


int main() {
    printf("Name - Rushi Daulatkar\n");
    printf("Roll No.:-53\n");
    struct Node *head = NULL;
    int choice, data;
```

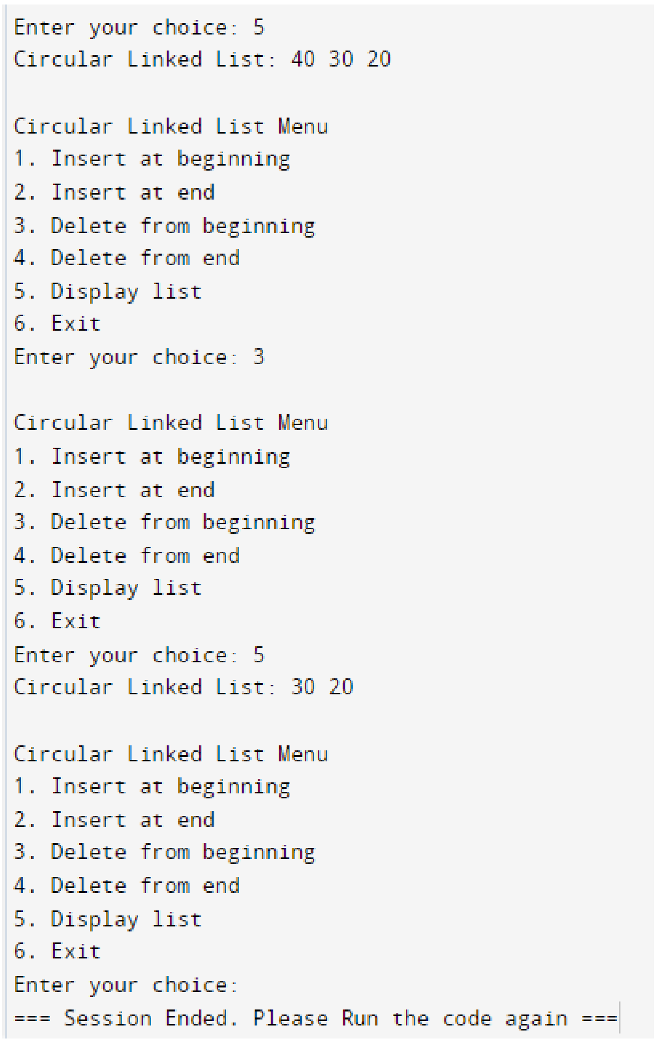
```
do {
    printf("\nCircular Linked List Menu\n");
    printf("1. Insert at beginning\n");
    printf("2. Insert at end\n");
    printf("3. Delete from beginning\n");
    printf("4. Delete from end\n");
    printf("5. Display list\n");
    printf("6. Exit\n");
    printf("Enter your choice: ");
    scanf("%d", &choice);

    switch (choice) {
        case 1:
            printf("Enter data to insert at beginning: ");
            scanf("%d", &data);
            insertAtStart(&head, data);
            break;
        case 2:
            printf("Enter data to insert at end: ");
            scanf("%d", &data);
            insertAtEnd(&head, data);
            break;
        case 3:
            deleteFromStart(&head);
            break;
        case 4:
            deleteFromEnd(&head);
            break;
        case 5:
            printf("Circular Linked List: ");
            display(head);
            break;
        case 6:
            printf("Exiting...\n");
            break;
        default:
            printf("Invalid choice! Please enter a valid option.\n");
        }
    } while (choice != 6);


    return 0;
}


                                                /*Function to insert a node
                                        at the beginning of the circular linked list.
                                                                          */
void insertAtStart(struct Node **head_ref, int new_data) {
    struct Node *new_node = (struct Node *)malloc(sizeof(struct Node));
    struct Node *last = *head_ref;
    new_node->data = new_data;


    if (*head_ref == NULL) {
        new_node->next = new_node;
    } else {
        while (last->next != *head_ref)
            last = last->next;


        last->next = new_node;
        new_node->next = *head_ref;
    }


    *head_ref = new_node;
}


                                        /* Function to insert a node
                                at the end of the circular linked list.*/
void insertAtEnd(struct Node **head_ref, int new_data) {
    struct Node *new_node = (struct Node *)malloc(sizeof(struct Node));
    struct Node *temp = *head_ref;


    new_node->data = new_data;
    new_node->next = *head_ref;


    if (*head_ref != NULL) {
        while (temp->next != *head_ref)

            temp = temp->next;

        temp->next = new_node;

    } else {

        new_node->next = new_node;

    }


    *head_ref = new_node;

}


                                                /*Function to delete a node from
                                           the beginning of the circular linked list*/

void deleteFromStart(struct Node **head_ref) {

    if (*head_ref == NULL) {

        printf("List is empty. Nothing to delete.\n");

        return;

    }


    struct Node *temp = *head_ref;

    struct Node *last = *head_ref;


    while (last->next != *head_ref)

        last = last->next;


    if (*head_ref == last) {
        free(*head_ref);
        *head_ref = NULL;
    } else {
        *head_ref = temp->next;
        last->next = temp->next;
        free(temp);
    }
}

                                          /* Function to delete a node from
                                          the end of the circular linked list*/
void deleteFromEnd(struct Node **head_ref) {
    if (*head_ref == NULL) {
        printf("List is empty. Nothing to delete.\n");
        return;
    }

    struct Node *temp = *head_ref;
    struct Node *last = *head_ref;

    while (last->next != *head_ref) {
        temp = last;
        last = last->next;
    }
    if (last == *head_ref) {
        free(*head_ref);
        *head_ref = NULL;
    } else {
        temp->next = last->next;
        free(last);
    }
}

                                                /* Function to display the circular
                                                            linked list*/
void display(struct Node *head) {
    struct Node *temp = head;
    if (head != NULL) {
        do {
            printf("%d ", temp->data);
            temp = temp->next;
        } while (temp != head);
    }
    printf("\n");
}
```

OUTPUT "-



```
/tmp/XhJ9JBxVqx.o
Name - Rushi Daulatkar
Roll No.:-53

Circular Linked List Menu
1. Insert at beginning
2. Insert at end
3. Delete from beginning
4. Delete from end
5. Display list
6. Exit
Enter your choice: 1
Enter data to insert at beginning: 10

Circular Linked List Menu
1. Insert at beginning
2. Insert at end
3. Delete from beginning
4. Delete from end
5. Display list
6. Exit
Enter your choice: 1
Enter data to insert at beginning: 20

Circular Linked List Menu
1. Insert at beginning
2. Insert at end
3. Delete from beginning
4. Delete from end
5. Display list
6. Exit
Enter your choice: 1
Enter data to insert at beginning: 30
```

```
Circular Linked List Menu
1. Insert at beginning
2. Insert at end
3. Delete from beginning
4. Delete from end
5. Display list
6. Exit
Enter your choice: 1
Enter data to insert at beginning: 40

Circular Linked List Menu
1. Insert at beginning
2. Insert at end
3. Delete from beginning
4. Delete from end
5. Display list
6. Exit
Enter your choice: 5
Circular Linked List: 40 30 20 10

Circular Linked List Menu
1. Insert at beginning
2. Insert at end
3. Delete from beginning
4. Delete from end
5. Display list
6. Exit
Enter your choice: 4

Circular Linked List Menu
1. Insert at beginning
2. Insert at end
3. Delete from beginning
4. Delete from end
5. Display list
```



```
Enter your choice: 5
Circular Linked List: 40 30 20

Circular Linked List Menu
1. Insert at beginning
2. Insert at end
3. Delete from beginning
4. Delete from end
5. Display list
6. Exit
Enter your choice: 3

Circular Linked List Menu
1. Insert at beginning
2. Insert at end
3. Delete from beginning
4. Delete from end
5. Display list
6. Exit
Enter your choice: 5
Circular Linked List: 30 20

Circular Linked List Menu
1. Insert at beginning
2. Insert at end
3. Delete from beginning
4. Delete from end
5. Display list
6. Exit
Enter your choice:
=== Session Ended. Please Run the code again ===
```