DATA STRUCTURE PRACTICAL NO. :-07

Aim :- Implement a Stack and perform the stack operations: Infix to Postfix, Infix to Prefix, Evaluation of Postfix Expression, Print using Menu Driver Program such as 1. Infix to Postfix, 2. Infix to Prefix, and 3. Evaluation of Postfix Expression and 4. Exit..

PROGARM :-

```
// circular linklist

#include <stdio.h>

#include <stdlib.h>


struct node {

    int data;

    struct node *next;

};


struct node *s;


void create() {

    struct node *p, *q;

    int ch;

    p = (struct node *)malloc(sizeof(struct node));

    printf("Enter the data of the first node\n");

    scanf("%d", &p->data);

    s = p;

    do {
        q = (struct node *)malloc(sizeof(struct node));
        printf("Enter the data of the next node\n");
        scanf("%d", &q->data);
        p->next = q;
        p = q;
        printf("\nPress 1 for the next node :\n");
        scanf("%d", &ch);
    } while (ch == 1);
    p->next = s;
}

void insert_beg() {
    struct node *x, *p;
    x = (struct node *)malloc(sizeof(struct node));
    printf("Enter the data of new node\n");
    scanf("%d", &x->data);
    if (s == NULL) {
        x->next = x;
        s = x;
    } else {
        p = s;
        while (p->next != s) {
            p = p->next;
        }
        x->next = s;
        p->next = x;
        s = x;
    }
}

void insert_end() {
    struct node *x, *p;
    x = (struct node *)malloc(sizeof(struct node));
    printf("Enter the data of new node\n");
    scanf("%d", &x->data);
    if (s == NULL) {
        x->next = x;
        s = x;
    } else {
        p = s;
        while (p->next != s) {
            p = p->next;
        }
        x->next = s;
        p->next = x;
    }
}
```

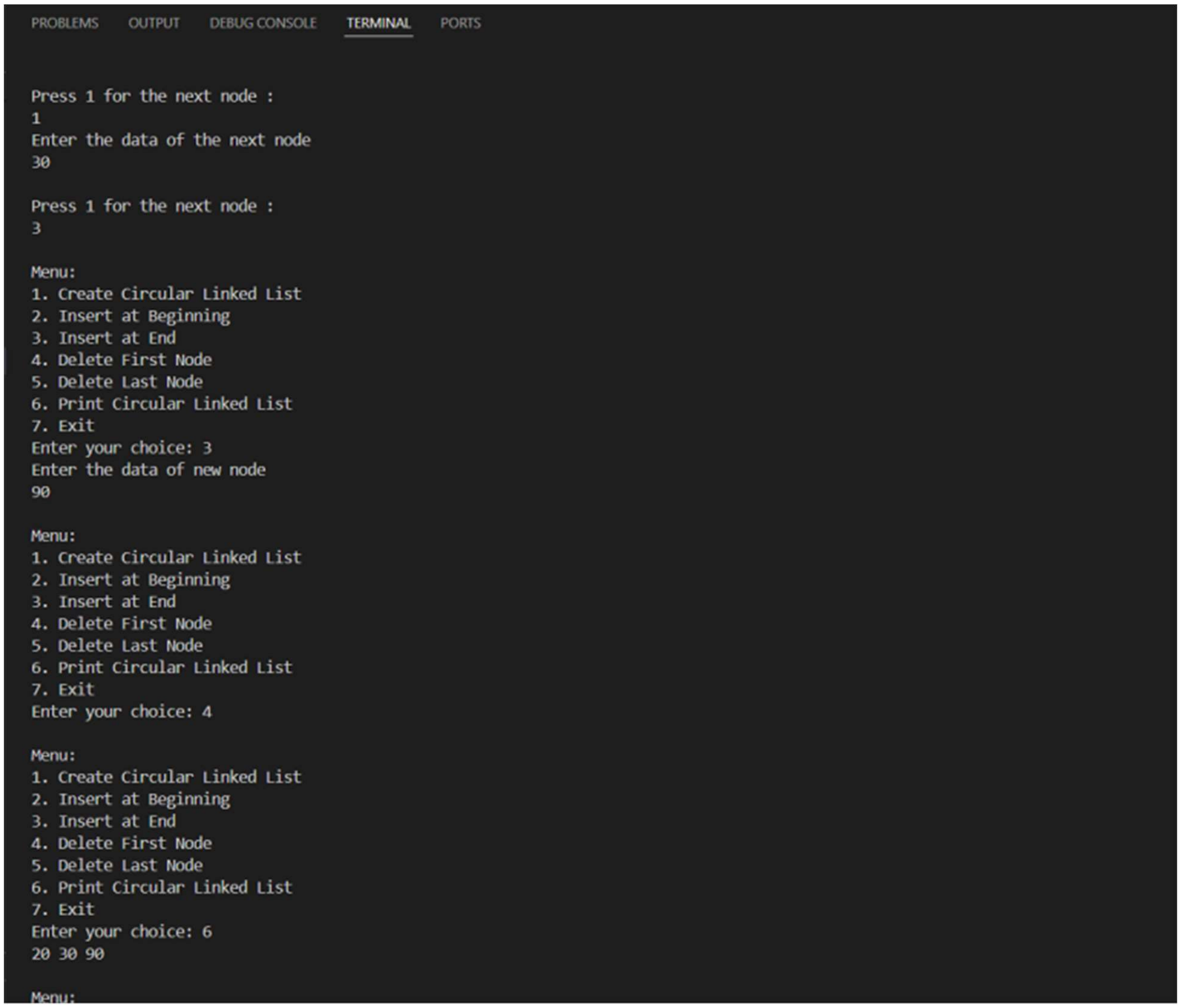
```
void delete_first() {
    struct node *q;
    if (s == NULL) {
        printf("Circular linked list is empty. Cannot delete.\n");
        return;
    }
    q = s;
    while (q->next != s) {
        q = q->next;
    }
    if (q == s) {
        free(s);
        s = NULL;
    } else {
        q->next = s->next;
        free(s);
        s = q->next;
    }
}

void delete_last() {
    struct node *p = s, *q = NULL;
    if (s == NULL) {
        printf("Circular linked list is empty. Cannot delete.\n");
        return;
    }
    while (p->next != s) {
        q = p;
        p = p->next;
    }
    if (q == NULL) {
        free(p);
        s = NULL;
    } else {
        q->next = s;
        free(p);
    }
}

void printCircularList() {
    if (s == NULL) {
        printf("Circular linked list is empty.\n");
        return;
    }
    struct node *p = s;
    do {
        printf("%d ", p->data);
        p = p->next;
    } while (p != s);
    printf("\n");
}

int main() {
    int choice;
    do {
        printf("\nMenu:\n");
        printf("1. Create Circular Linked List\n");
        printf("2. Insert at Beginning\n");
        printf("3. Insert at End\n");
        printf("4. Delete First Node\n");
        printf("5. Delete Last Node\n");
        printf("6. Print Circular Linked List\n");
        printf("7. Exit\n");
        printf("Enter your choice: ");
        scanf("%d", &choice);
        switch (choice) {
            case 1:
                create();
                break;
            case 2:
                insert_beg();
                break;
            case 3:
                insert_end();
                break;
            case 4:
                delete_first();
                break;
            case 5:
                delete_last();
                break;
            case 6:
                printCircularList();
                break;
            case 7:
                printf("Exiting...\n");
                break;
            default:
                printf("Invalid choice. Please try again.\n");
        }
    } while (choice != 7);

    return 0;
}
```



```
Press 1 for the next node :
1
Enter the data of the next node
30

Press 1 for the next node :
3

Menu:
1. Create Circular Linked List
2. Insert at Beginning
3. Insert at End
4. Delete First Node
5. Delete Last Node
6. Print Circular Linked List
7. Exit
Enter your choice: 3
Enter the data of new node
90

Menu:
1. Create Circular Linked List
2. Insert at Beginning
3. Insert at End
4. Delete First Node
5. Delete Last Node
6. Print Circular Linked List
7. Exit
Enter your choice: 4

Menu:
1. Create Circular Linked List
2. Insert at Beginning
3. Insert at End
4. Delete First Node
5. Delete Last Node
6. Print Circular Linked List
7. Exit
Enter your choice: 6
20 30 90

Menu:
```

GITHUB LINK OF PRACTICAL No. 07 :-
https://github.com/sidheshwar2005/Data_strucutre_practical.git