

WAP to Implement Circular Singly Linked List with following operations:-

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

typedef struct Node {
    int data;
    struct Node *link;
} Node;

Node *head = NULL;

Node* createNODE(int data) {
    Node *nn = (Node *)malloc(sizeof(Node));
    nn->data = data;
    nn->link = NULL;
    return nn;
}

void createList(int data) {
    Node *nn = createNODE(data);

    if (head == NULL) {
        head = nn;
        nn->link = head;
        return;
    }

    Node *temp = head;
    while (temp->link != head)
        temp = temp->link;

    temp->link = nn;
    nn->link = head;
}

void insertAtFirst(int data) {
    Node *nn = createNODE(data);

    if (head == NULL) {
        head = nn;
        nn->link = head;
        return;
    }

    Node *temp = head;
    while (temp->link != head)
        temp = temp->link;
```

```

nn->link = head;
temp->link = nn;
head = nn;
}

void insertAtEnd(int data) {
    createList(data);
}

void insertAtPosition(int data, int pos) {
    if (pos == 1) {
        insertAtFirst(data);
        return;
    }

    Node *nn = createNODE(data);
    Node *temp = head;

    for (int i = 1; i < pos - 1; i++) {
        if (temp->link == head) {
            printf("Position out of range");
            return;
        }
        temp = temp->link;
    }

    nn->link = temp->link;
    temp->link = nn;
}

void deleteAtFirst() {
    if (head == NULL)
        return;

    if (head->link == head) {
        free(head);
        head = NULL;
        return;
    }

    Node *temp = head;
    while (temp->link != head)
        temp = temp->link;

    Node *del = head;
    temp->link = head->link;
    head = head->link;
}

```

```

        free(del);
    }

void deleteAtEnd() {
    if (head == NULL)
        return;

    if (head->link == head) {
        free(head);
        head = NULL;
        return;
    }

    Node *temp = head;
    while (temp->link->link != head)
        temp = temp->link;

    Node *last = temp->link;
    temp->link = head;
    free(last);
}

void deleteAtPosition(int pos) {
    if (head == NULL)
        return;

    if (pos == 1) {
        deleteAtFirst();
        return;
    }

    Node *temp = head;

    for (int i = 1; i < pos - 1; i++) {
        if (temp->link == head) {
            printf("Position out of range.");
            return;
        }
        temp = temp->link;
    }

    Node *del = temp->link;

    if (del == head) {
        printf("Position out of range.");
        return;
    }
}

```

```

temp->link = del->link;
free(del);
}

void display() {
    if (head == NULL) {
        printf("List is empty");
        return;
    }

    Node *temp = head;
    do {
        printf("%d -> ", temp->data);
        temp = temp->link;
    } while (temp != head);

    printf("(head)\n");
}

int main() {
    FILE *fp = fopen("input.txt", "r");
    if (fp == NULL) {
        printf("file cant open please check :)");
        return 0;
    }
    printf("1. Insert at End (Create List)\n");
    printf("2. Insert at First\n");
    printf("3. Insert at End\n");
    printf("4. Insert at Position\n");
    printf("5. Delete at First\n");
    printf("6. Delete at End\n");
    printf("7. Delete at Position\n");
    printf("8. Display List\n");
    printf("9. Exit\n");

    int choice, data, pos;
    while (fscanf(fp, "%d", &choice) != EOF) {
        switch (choice) {
            case 1:
                fscanf(fp, "%d", &data);
                createList(data);
                break;

            case 2:
                fscanf(fp, "%d", &data);
                insertAtFirst(data);
                break;
        }
    }
}

```

```
case 3:  
    fscanf(fp, "%d", &data);  
    insertAtEnd(data);  
    break;  
  
case 4:  
    fscanf(fp, "%d %d", &data, &pos);  
    insertAtPosition(data, pos);  
    break;  
  
case 5:  
    deleteAtFirst();  
    break;  
  
case 6:  
    deleteAtEnd();  
    break;  
  
case 7:  
    fscanf(fp, "%d", &pos);  
    deleteAtPosition(pos);  
    break;  
  
case 8:  
    display();  
    break;  
  
case 9:  
    fclose(fp);  
    exit(0);  
  
default:  
    printf("Invalid choice. Please correct the input file");  
}  
}  
  
fclose(fp);  
return 0;  
}
```

INPUT FILE:-

```
File Edit View
```

```
1 10
1 20
2 5
3 40
4 25 3
8
9
```

OUTPUT:-

```
5 -> 10 -> 25 -> 20 -> 40 -> (head)
PS C:\Users\mahme\Desktop\C_prms\bmsce_LaPro> gcc CLL.c -o CLL
PS C:\Users\mahme\Desktop\C_prms\bmsce_LaPro> ./CLL
1. Insert at End (Create List)
2. Insert at First
3. Insert at End
4. Insert at Position
5. Delete at First
6. Delete at End
7. Delete at Position
8. Display List
9. Exit
5 -> 10 -> 25 -> 20 -> 40 -> (head)
PS C:\Users\mahme\Desktop\C_prms\bmsce_LaPro>
```