dequeue-using-linked-list.c

```
//implement double ended queue using linked list.
#include <stdio.h>
#include <stdlib.h>

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

typedef struct DeQueue{
   node *front;
   node *rear;
}dequeue;

dequeue create(){
   dequeue dq = {NULL, NULL};
   return dq;
}

int rear(dequeue dq){
   return dq.rear->data;
}

int front(dequeue dq){
   return dq.front->data;
}
int len(dequeue dq){
   if(dq.front == NULL){
      return 0;
   }
   int count = 1;
   while(dq.rear != dq.front){
      count++;
      dq.rear = dq.rear->next;
   }
   return count;
}
int isEmpty(dequeue dq){
   if(dq.front){
      return 0;
   }
   return 1;
}

void enq_rear(dequeue *dq, int new_data){
   node *new_node = (node*)malloc(sizeof(node));
   new_node->data = new_data;
   new_node->next = dq->rear;
   if(dq->rear == NULL){
      dq->front = new_node;
   }
   dq->rear = new_node;
}

void enq_front(dequeue *dq, int new_data){
   node *new_node = (node*)malloc(sizeof(node));
   new_node->data = new_data;
   new_node->next = NULL;
   dq->front->next = new_node;
   if(dq->front == NULL){
      dq->rear = new_node;
   }
   dq->front = new_node;
}
int deq_rear(dequeue *dq){
```

```
    if(!dq->rear){
        printf("UnderFlow.\n");
        return -1;
    }
    node *temp = dq->rear;
    dq->rear = dq->rear->next;
    int data = temp->data;
    free(temp);
    return data;
}

int deq_front(dequeue *dq){
    if(dq->front == NULL){
        printf("Underflow.\n");
        return -1;
    }
    int data;
    node *temp = dq->rear;
    if(dq->rear == dq->front){
        data = temp->data;
        *dq = create();
        free(temp);
        return data;
    }
    while(temp->next != dq->front){
        temp = temp->next;
    }
    temp->next = NULL;
    node *next = dq->front;
    dq->front = temp;
    data = next->data;
    free(next);
    return data;
}

void display(dequeue dq){
    if(dq.rear == NULL){
        printf("The Queue is Empty.");
    }
    node *temp = dq.rear;
    while(temp != NULL){
        printf("%d",temp->data);
        temp = temp->next;
        if(temp != NULL){
            printf(" -> ");
        }
    }
    printf("\n");
}

void main(){
    int data;
    int op;
    dequeue dq = create();
    while(1){
        printf("1. enQueueR, 2. enQueueF, 3. deQueueR, 4. deQueueF, 5. Display, 6. Size, 7. Rear, 8. Front, 9. Exit.\nEnter Your Choice: ");
        scanf("%d",&op);
        switch(op){
            case 1:
                printf("Enter the element: ");
                scanf("%d",&data);
                enq_rear(&dq, data);
                break;
            case 2:
                printf("Enter the element: ");
                scanf("%d",&data);
```

```
            enq_front(&dq, data);
            break;
        case 3:
            printf("Removed Rear %d\n",deq_rear(&dq));
            break;
        case 4:
            printf("Removed Front %d\n",deq_front(&dq));
            break;
        case 5:
            display(dq);
            break;
        case 6:
            printf("The len of this Queue is %d\n",len(dq));
            break;
        case 7:
            printf("The Rear of this Queue is %d\n",rear(dq));
            break;
        case 8:
            printf("The Front of this Queue is %d\n",front(dq));
            break;
        case 9:
            printf("Oops..");
            return;
        default:
            printf("Wrong Input.\n");
    }
  }
}
```

OUTPUT

```
PS S:\WorkSpace\CollegeWork\DataStructure> gcc .\dequeue-using-linekd-list.c
PS S:\WorkSpace\CollegeWork\DataStructure> ./a
1. enQueueR, 2. enQueueF, 3. deQueueR, 4. deQueueF, 5. Display, 6. Size, 7. Rear, 8. Front, 9. Exit.
Enter Your Choice: 1
Enter the element: 12
1. enQueueR, 2. enQueueF, 3. deQueueR, 4. deQueueF, 5. Display, 6. Size, 7. Rear, 8. Front, 9. Exit.
Enter Your Choice: 1
Enter the element: 13
1. enQueueR, 2. enQueueF, 3. deQueueR, 4. deQueueF, 5. Display, 6. Size, 7. Rear, 8. Front, 9. Exit.
Enter Your Choice: 2
Enter the element: 23
1. enQueueR, 2. enQueueF, 3. deQueueR, 4. deQueueF, 5. Display, 6. Size, 7. Rear, 8. Front, 9. Exit.
Enter Your Choice: 5
13 -> 12 -> 23
1. enQueueR, 2. enQueueF, 3. deQueueR, 4. deQueueF, 5. Display, 6. Size, 7. Rear, 8. Front, 9. Exit.
Enter Your Choice: 2
Enter the element: 34
1. enQueueR, 2. enQueueF, 3. deQueueR, 4. deQueueF, 5. Display, 6. Size, 7. Rear, 8. Front, 9. Exit.
Enter Your Choice: 5
13 -> 12 -> 23 -> 34
1. enQueueR, 2. enQueueF, 3. deQueueR, 4. deQueueF, 5. Display, 6. Size, 7. Rear, 8. Front, 9. Exit.
Enter Your Choice: 3
Removed Rear 13
1. enQueueR, 2. enQueueF, 3. deQueueR, 4. deQueueF, 5. Display, 6. Size, 7. Rear, 8. Front, 9. Exit.
Enter Your Choice: 4
Removed Front 34
1. enQueueR, 2. enQueueF, 3. deQueueR, 4. deQueueF, 5. Display, 6. Size, 7. Rear, 8. Front, 9. Exit.
Enter Your Choice: 5
12 -> 23
1. enQueueR, 2. enQueueF, 3. deQueueR, 4. deQueueF, 5. Display, 6. Size, 7. Rear, 8. Front, 9. Exit.
Enter Your Choice: 9
Oops..
PS S:\WorkSpace\CollegeWork\DataStructure>
```