Implementation of Queue using Singly linked list

```
// code
#include <stdio.h>
#include <stdlib.h>
// Implementation of queue with liked list

// Declaration of node of linked list
struct node {
   int data;
   struct node *next;
};

// front of linked list
struct node *front = NULL;

// rear of linked list
struct node *rear = NULL;


void insert(int val) { // insert value at front
   // declare and allocate memory of newNode
   struct node *newNode;
   newNode = (struct node *)malloc(sizeof(struct node));

   newNode->data = val;

   if (front == NULL && rear == NULL) {    // when first node is added
      front = newNode;
      rear = newNode;
      newNode->next = NULL;
   }
   else { // insertion of any other node
      rear->next = newNode;
      rear = newNode;
      newNode->next = NULL;
   }
}

void delete () {    // deletes node at front
   // traversing pointer
   struct node *ptr;
   ptr = front;

   if (front == NULL && rear == NULL) {     // checks if queue is empty
      printf("\nQueue is empty!");
      return;
   }

   // moves 'front' ahead
   front = ptr->next;
   printf("\nDeleted element is :  %d", ptr->data);
```

```
    free(ptr);

    if (front == NULL) {    // when last node is deleted
        rear = NULL;
    }
}

void showFront() {    //displays element at front

    if (front == NULL && rear == NULL) {     // checks if queue is empty
        printf("\nQueue is empty!");
        return;
    }

    // displays element at front
    printf("\nElement at front is :  %d", front->data);
}

int size() {    // returns size of the queue

    if (front == NULL && rear == NULL) {     // if queue is empty
        return 0;
    }
    int count = 1;

    // traversing pointer
    struct node *ptr;
    ptr = front;
    while (ptr->next != NULL) {    // count number of nodes in queue
        ptr = ptr->next;
        count++;
    }
    return count;
}

void display() {    // display elements of queue
    // traversing pointer
    struct node *ptr;
    ptr = front;

    if (front == NULL && rear == NULL) {     // checks if queue is empty
        printf("\nQueue is empty!");
        return;
    }
    printf("\nElements in queue are :  ");
    while (ptr->next != NULL) {    // traverse and display
        printf("%d  ", ptr->data);
        ptr = ptr->next;
    }
    printf("%d", ptr->data);
}
```

```
int main() {

    int choice, val;

    while (1) {
        printf("\n*1. INSERT");
        printf("\n*2. DELETE");
        printf("\n*3. SHOW FRONT");
        printf("\n*4. SIZE");
        printf("\n*5. DISPLAY");
        printf("\n*6. EXIT");
        printf("\nEnter your choice :  ");
        scanf("%d", &choice);

        switch (choice) {

        case 1:
            printf("\nEnter an element to insert :  ");
            scanf("%d", &val);
            insert(val);
            break;
        case 2:
            delete();
            break;
        case 3:
            showFront();
            break;
        case 4:
            printf("\nSize of queue is :  %d", size());
            break;
        case 5:
            display();
            break;
        case 6:
            printf("\n *** E X I T I N G ***");
            exit(1);
        default:
            printf("\nINVALID INPUT");
        }
    }
    return 0;
}
```

// Output

```
*1. INSERT
*2. DELETE
*3. SHOW FRONT
*4. SIZE
*5. DISPLAY
*6. EXIT
Enter your choice :  1

Enter an element to insert :  5

*1. INSERT
*2. DELETE
*3. SHOW FRONT
*4. SIZE
*5. DISPLAY
*6. EXIT
Enter your choice :  1

Enter an element to insert :  10

*1. INSERT
*2. DELETE
*3. SHOW FRONT
*4. SIZE
*5. DISPLAY
*6. EXIT
Enter your choice :  1

Enter an element to insert :  15

*1. INSERT
*2. DELETE
*3. SHOW FRONT
*4. SIZE
*5. DISPLAY
*6. EXIT
Enter your choice :  4

Size of queue is :  3
*1. INSERT
*2. DELETE
*3. SHOW FRONT
*4. SIZE
*5. DISPLAY
*6. EXIT
Enter your choice :  5

Elements in queue are :  5  10  15
```

```
*1. INSERT
*2. DELETE
*3. SHOW FRONT
*4. SIZE
*5. DISPLAY
*6. EXIT
Enter your choice :  2

Deleted element is :  5
*1. INSERT
*2. DELETE
*3. SHOW FRONT
*4. SIZE
*5. DISPLAY
*6. EXIT
Enter your choice :  3

Element at front is :  10
*1. INSERT
*2. DELETE
*3. SHOW FRONT
*4. SIZE
*5. DISPLAY
*6. EXIT
Enter your choice :  2

Deleted element is :  10
*1. INSERT
*2. DELETE
*3. SHOW FRONT
*4. SIZE
*5. DISPLAY
*6. EXIT
Enter your choice :  5

Elements in queue are :  15
*1. INSERT
*2. DELETE
*3. SHOW FRONT
*4. SIZE
*5. DISPLAY
*6. EXIT
Enter your choice :  2

Deleted element is :  15
```

```
*1. INSERT
*2. DELETE
*3. SHOW FRONT
*4. SIZE
*5. DISPLAY
*6. EXIT
Enter your choice :  1

Enter an element to insert :  200

*1. INSERT
*2. DELETE
*3. SHOW FRONT
*4. SIZE
*5. DISPLAY
*6. EXIT
Enter your choice :  5

Elements in queue are :  200
*1. INSERT
*2. DELETE
*3. SHOW FRONT
*4. SIZE
*5. DISPLAY
*6. EXIT
Enter your choice :  6

 *** E X I T I N G ***
```