Name: KAMCHE YANN ARNAUD

Matricule : FE21A208

Department: Computer Engineering

Level: 300

Task: Implement a queue using Linked List

1. CODE

```
/* Implementation of a Queue using Link List
Author: Kamche Yann Arnaud
Date: 12/05/2022
*/
#include<stdio.h>
#include<stdlib.h>
#include<conio.h>
struct Node{
   int data;
   struct Node* next;
};
struct Node* front = NULL;
struct Node* rear = NULL;

//Queue is empty
void emptyQueue(){
   if (front == NULL && rear == NULL)
         printf("NULL");
         return;
}
//Enqueue enters an element into the queue
void Enqueue(int x){
   struct Node* temp = (struct Node*)malloc(sizeof(struct Node*));
   temp->data = x;
   temp->next = NULL;
   if(front == NULL && rear == NULL){
         front = rear = temp;
         return;
   }
   rear->next = temp;
   rear = temp;
}
//Dequeue removes an element from the queue
void Dequeue(){
         struct Node* temp = (struct Node*)malloc(sizeof(struct Node*));
         if (front == NULL){
               emptyQueue();
               return;
         }
         if(front == rear){
               front = rear = NULL;
         }
         else{
               front = front->next;
         }
         free(temp);
}
//Display prints the element of the queue
void Display(){
   struct Node* temp = (struct Node*)malloc(sizeof(struct Node*));
   printf("front -> ");
   if(front == NULL && rear == NULL){
         printf("NULL");
         return;
   }

   else{
```

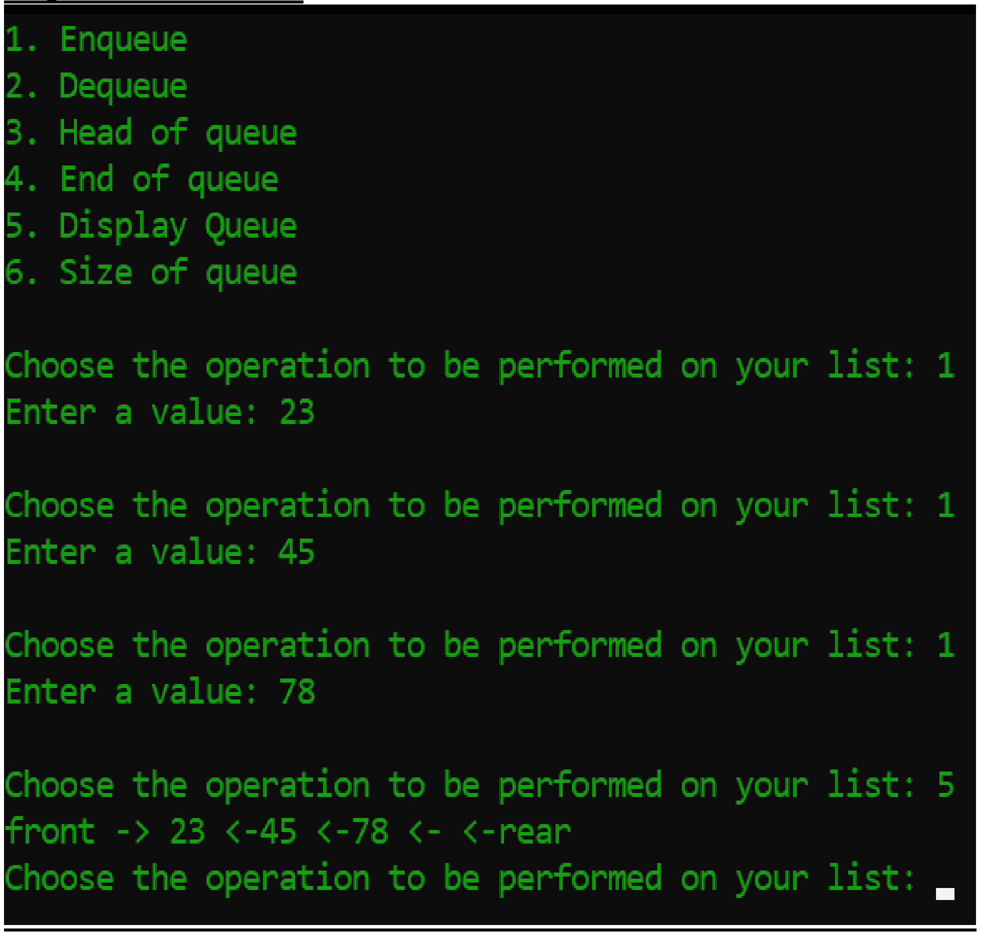
```
            temp = front;
            while(temp != NULL){
                  printf("%d <-", temp->data);
                  temp = temp->next;
            }
      printf(" <-rear");
      return;
      }
   }
   //Front of the queue
   int headOfQueue(){
      if(front == NULL)
            return -1;
      else
            return front->data;
   }
   //Rear of the queue
   int endOfQueue(){
      if(rear == NULL)
            return -1;
      else
            return rear->data;
   }
   //size of the Queue
   int sizeOfQueue(){
      int count = 0;
      struct Node* temp = (struct Node*)malloc(sizeof(struct Node*));
      temp = front;
      while(temp!= NULL){
            count++;
            temp = temp->next;
      }
      return count;
```

```
}
int main(){
   system("color 2");
   int choice, num;
   printf("1. Enqueue\n");
   printf("2. Dequeue\n ");
   printf("3. Head of queue\n");
   printf("4. End of queue\n");
   printf("5. Display Queue\n");
   printf("6. Size of queue\n");
   options:
   printf("\nChoose the operation to be performed on your list: ");
   scanf("%d", &choice);
   switch(choice){
         case 1:
               printf("Enter a value: ");
               scanf("%d", &num);
               Enqueue(num);
               break;
         case 2:
               Dequeue();
               break;
         case 3:
               if(headOfQueue() == -1)
                     printf("head: NULL");
               else
                     printf("head: %d", headOfQueue());
               break;
         case 4:
               if(endOfQueue() == -1)
                     printf("end: NULL");
               else
                     printf("end: %d", endOfQueue());
```

```
            break;
        case 5:
            Display();
            break;
        case 6:
            printf("%d", sizeOfQueue());
            break;
        default:
            break;
    }
    goto options;
    return 0;
}
```

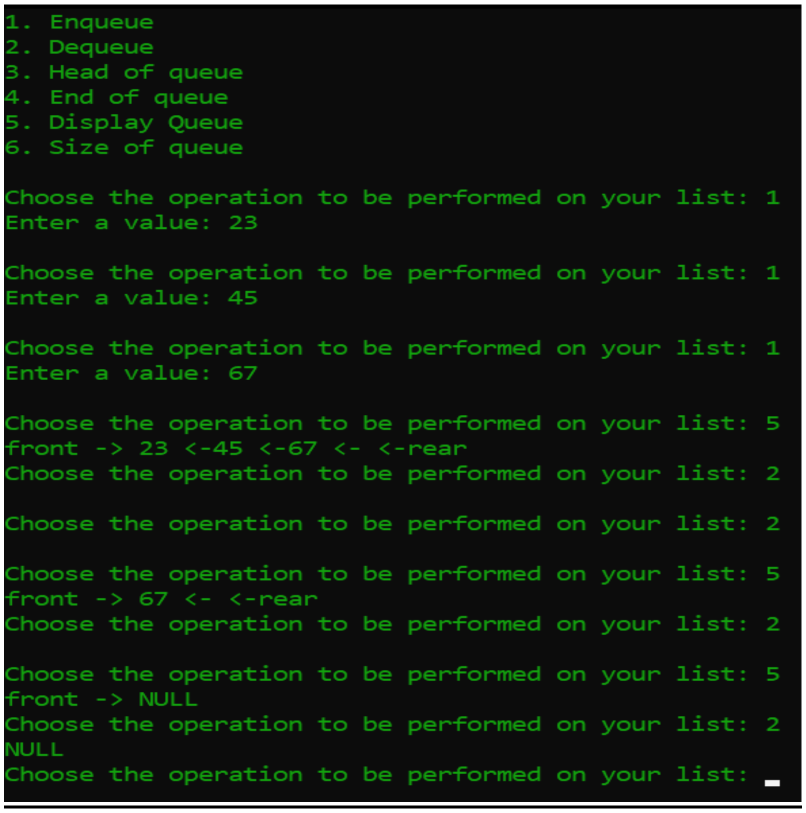
1. **COMPILATION RESULTS**

I) **ENQUEUE OPERATION**



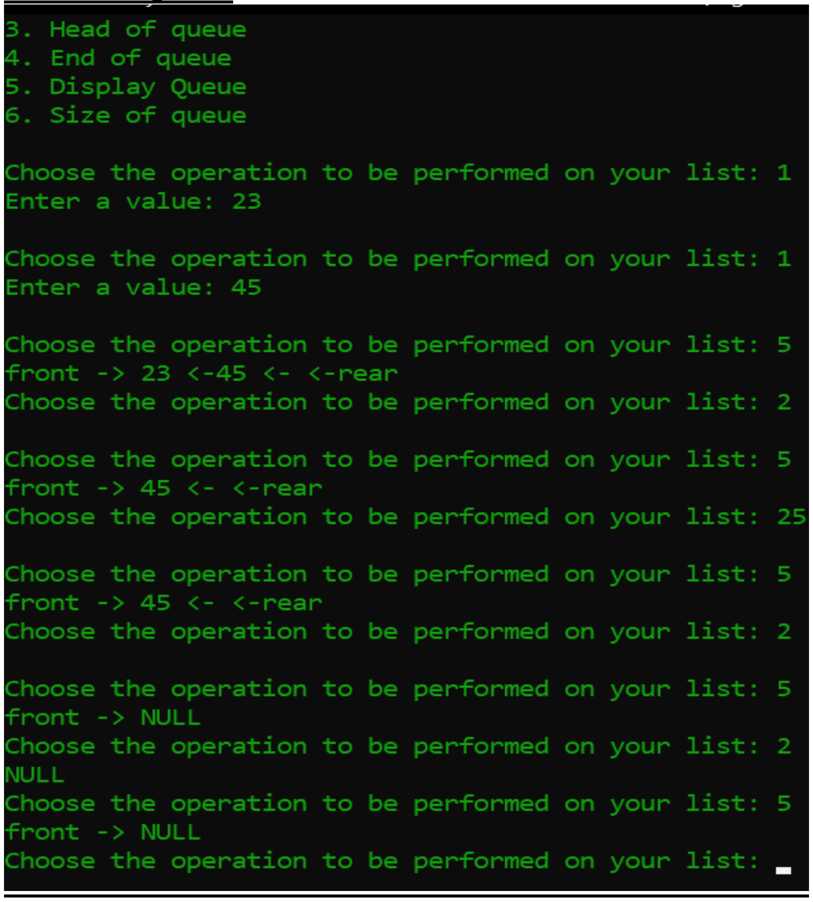
```
1. Enqueue
2. Dequeue
3. Head of queue
4. End of queue
5. Display Queue
6. Size of queue

Choose the operation to be performed on your list: 1
Enter a value: 23

Choose the operation to be performed on your list: 1
Enter a value: 45

Choose the operation to be performed on your list: 1
Enter a value: 78

Choose the operation to be performed on your list: 5
front -> 23 <-45 <-78 <- <-rear
Choose the operation to be performed on your list:
```

II) DEQUEUE OPERATION



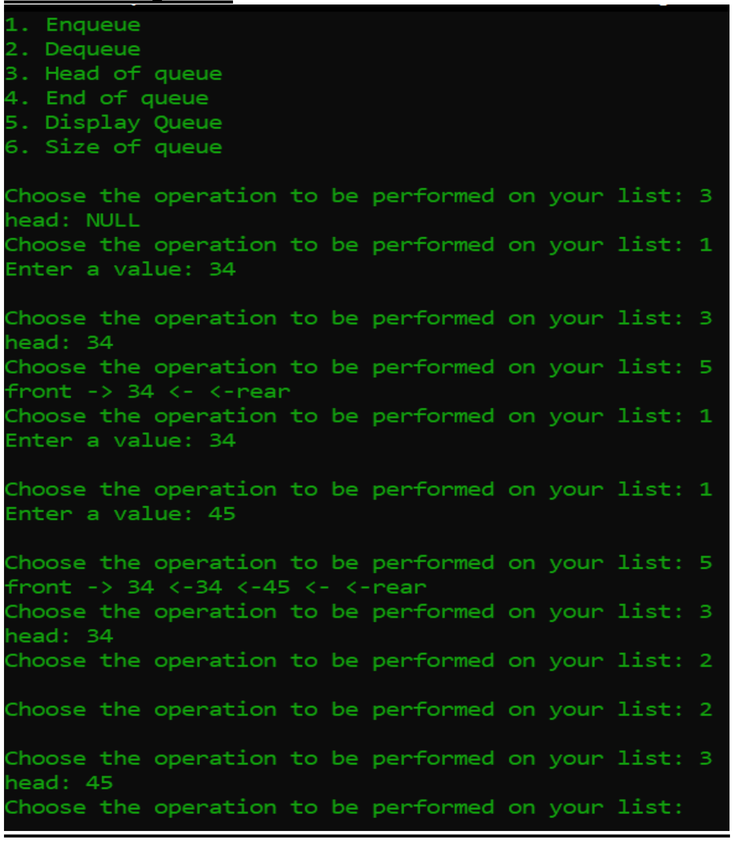
```
1. Enqueue
2. Dequeue
3. Head of queue
4. End of queue
5. Display Queue
6. Size of queue

Choose the operation to be performed on your list: 1
Enter a value: 23

Choose the operation to be performed on your list: 1
Enter a value: 45

Choose the operation to be performed on your list: 1
Enter a value: 67

Choose the operation to be performed on your list: 5
front -> 23 <-45 <-67 <- <-rear
Choose the operation to be performed on your list: 2

Choose the operation to be performed on your list: 2

Choose the operation to be performed on your list: 5
front -> 67 <- <-rear
Choose the operation to be performed on your list: 2

Choose the operation to be performed on your list: 5
front -> NULL
Choose the operation to be performed on your list: 2
NULL
Choose the operation to be performed on your list:
```

III) DISPLAY QUEUE



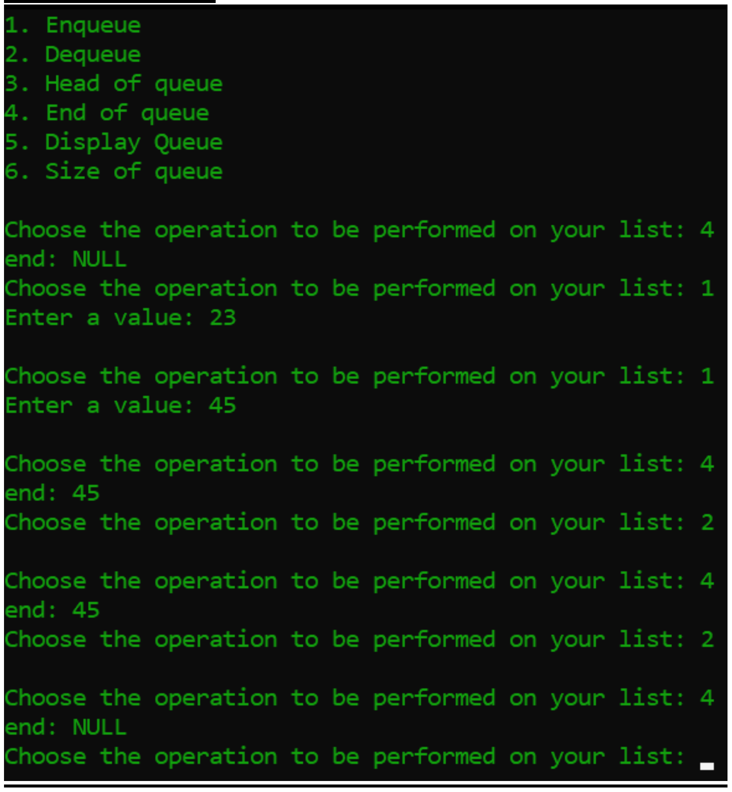
```
3. Head of queue
4. End of queue
5. Display Queue
6. Size of queue

Choose the operation to be performed on your list: 1
Enter a value: 23

Choose the operation to be performed on your list: 1
Enter a value: 45

Choose the operation to be performed on your list: 5
front -> 23 <-45 <- <-rear
Choose the operation to be performed on your list: 2

Choose the operation to be performed on your list: 5
front -> 45 <- <-rear
Choose the operation to be performed on your list: 25

Choose the operation to be performed on your list: 5
front -> 45 <- <-rear
Choose the operation to be performed on your list: 2

Choose the operation to be performed on your list: 5
front -> NULL
Choose the operation to be performed on your list: 2
NULL
Choose the operation to be performed on your list: 5
front -> NULL
Choose the operation to be performed on your list:
```

IV) HEAD OF QUEUE



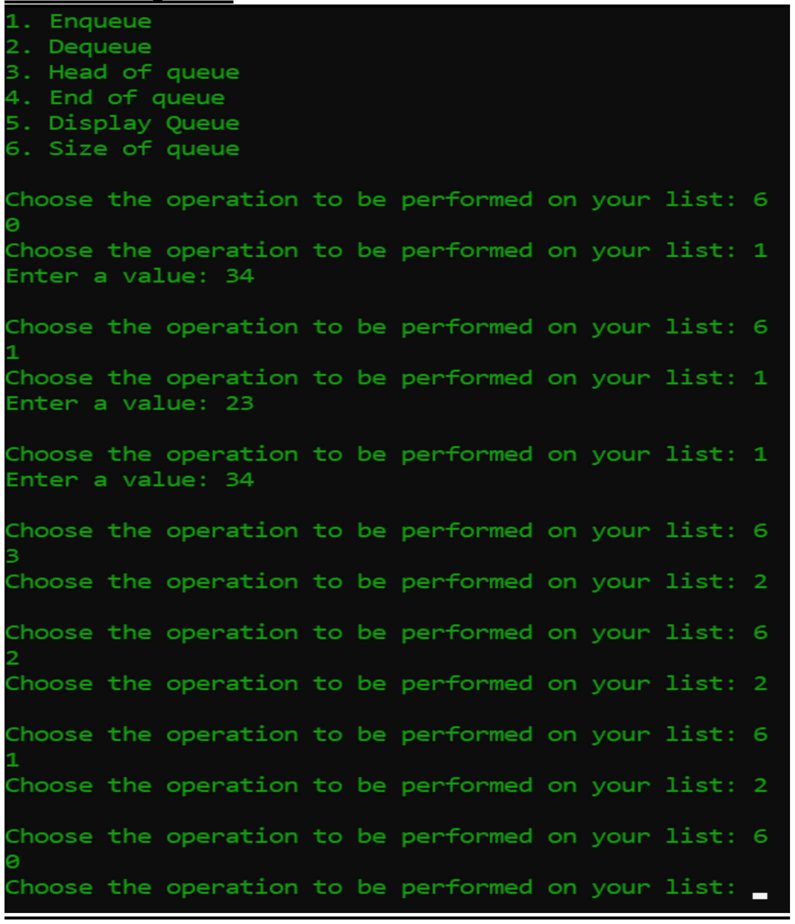
```
1. Enqueue
2. Dequeue
3. Head of queue
4. End of queue
5. Display Queue
6. Size of queue

Choose the operation to be performed on your list: 3
head: NULL
Choose the operation to be performed on your list: 1
Enter a value: 34

Choose the operation to be performed on your list: 3
head: 34
Choose the operation to be performed on your list: 5
front -> 34 <- <-rear
Choose the operation to be performed on your list: 1
Enter a value: 34

Choose the operation to be performed on your list: 1
Enter a value: 45

Choose the operation to be performed on your list: 5
front -> 34 <-34 <-45 <- <-rear
Choose the operation to be performed on your list: 3
head: 34
Choose the operation to be performed on your list: 2

Choose the operation to be performed on your list: 2

Choose the operation to be performed on your list: 3
head: 45
Choose the operation to be performed on your list:
```

V) END OF QUEUE



```
1. Enqueue
2. Dequeue
3. Head of queue
4. End of queue
5. Display Queue
6. Size of queue

Choose the operation to be performed on your list: 4
end: NULL
Choose the operation to be performed on your list: 1
Enter a value: 23

Choose the operation to be performed on your list: 1
Enter a value: 45

Choose the operation to be performed on your list: 4
end: 45
Choose the operation to be performed on your list: 2

Choose the operation to be performed on your list: 4
end: 45
Choose the operation to be performed on your list: 2

Choose the operation to be performed on your list: 4
end: NULL
Choose the operation to be performed on your list:
```

VI) **<u>SIZE OF QUEUE</u>**



```
1. Enqueue
2. Dequeue
3. Head of queue
4. End of queue
5. Display Queue
6. Size of queue

Choose the operation to be performed on your list: 6
0
Choose the operation to be performed on your list: 1
Enter a value: 34

Choose the operation to be performed on your list: 6
1
Choose the operation to be performed on your list: 1
Enter a value: 23

Choose the operation to be performed on your list: 1
Enter a value: 34

Choose the operation to be performed on your list: 6
3
Choose the operation to be performed on your list: 2

Choose the operation to be performed on your list: 6
2
Choose the operation to be performed on your list: 2

Choose the operation to be performed on your list: 6
1
Choose the operation to be performed on your list: 2

Choose the operation to be performed on your list: 6
0
Choose the operation to be performed on your list: _
```