DS Lab Assignment 5

Pranav Joshi
CS-B Batch 2
Roll no: 43

Title: WAP to implement Linear and Circular Queue using Array and Linked Lists.

Code:

1. Linear Queue using Array

```c
#include<stdio.h>
#define MAX 10

int queue[MAX];
int front = -1, rear = -1;

void enqueue(int x){
   if(rear == MAX-1){
      printf("\nQueue Overflow\n");
   }
   else{
      if(front == -1){
         front = 0;
      }
      rear++;
      queue[rear] = x;
      printf("\nInserted element is: %d",x);
   }
}

void dequeue(){
   if(front == -1 || front > rear){
      printf("\nQueue Underflow\n");
   }
   else{
      printf("\nDeleted element is: %d",queue[front]);
      front++;
   }
}

void display(){
```

```
   int i;
   if(front == -1){
      printf("\nQueue is empty\n");
   }
   else{
      printf("\nQueue elements are:\n");
      for(i=front; i<=rear; i++){
         printf("%d\t",queue[i]);
      }
   }
}

int main(){
   int ch, val;
   do{
      printf("\n1. Enqueue");
      printf("\n2. Dequeue");
      printf("\n3. Display");
      printf("\n4. Exit");
      printf("\nEnter your choice: ");
      scanf("%d",&ch);
      switch(ch){
         case 1:
            printf("\nEnter the element to be inserted: ");
            scanf("%d",&val);
            enqueue(val);
            break;
         case 2:
            dequeue();
            break;
         case 3:
            display();
            break;
         case 4:
            printf("\nExit\n");
            break;
         default:
            printf("\nInvalid Choice\n");
      }
   }while(ch!=4);
   return 0;
}
```

Output:

```
PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL

PS C:\Code\C\Code> cd "c:\Code\C\Code\" ; if ($?) { gcc QueueLinearArray.c -o QueueLinearArray } ; if ($?) { .\QueueLinearArray }

1. Enqueue
2. Dequeue
3. Display
4. Exit
Enter your choice: 1

Enter the element to be inserted: 10

Inserted element is: 10
1. Enqueue
2. Dequeue
3. Display
4. Exit
Enter your choice: 1

Enter the element to be inserted: 20

Inserted element is: 20
1. Enqueue
2. Dequeue
3. Display
4. Exit
Enter your choice: 1

Enter the element to be inserted: 30

Inserted element is: 30
1. Enqueue
2. Dequeue
3. Display
4. Exit
Enter your choice: 3

Queue elements are:
10      20      30
1. Enqueue
2. Dequeue
3. Display
4. Exit
Enter your choice: 2

Deleted element is: 10
1. Enqueue
2. Dequeue
3. Display
4. Exit
Enter your choice: 3

Queue elements are:
20      30
1. Enqueue
2. Dequeue
3. Display
4. Exit
Enter your choice: 4

Exit
PS C:\Code\C\Code>
```

2. Circular Queue using Array

```c
#include<stdio.h>
#define MAX 10

int queue[MAX];
int front = -1, rear = -1;
```

```
void enqueue(int x){
   if((front == 0 && rear == MAX-1) || (front == rear+1)){
      printf("\nQueue Overflow\n");
   }
   else{
      if(front == -1){
         front = 0;
      }
      rear = (rear+1)%MAX;
      queue[rear] = x;
      printf("\nInserted element is: %d",x);
   }
}

void dequeue(){
   if(front == -1){
      printf("\nQueue Underflow\n");
   }
   else{
      printf("\nDeleted element is: %d",queue[front]);
      if(front == rear){
         front = -1;
         rear = -1;
      }
      else{
         front = (front+1)%MAX;
      }
   }
}

void display(){
   int i;
   if(front == -1){
      printf("\nQueue is empty\n");
   }
   else{
      printf("\nQueue elements are:\n");
      if(front <= rear){
         for(i=front; i<=rear; i++){
            printf("%d\t",queue[i]);
         }
      }
      else{
         for(i=front; i<MAX; i++){
            printf("%d\t",queue[i]);
         }
         for(i=0; i<=rear; i++){
            printf("%d\t",queue[i]);
```

```
         }
      }
   }
}

int main(){
   int ch, val;
   do{
      printf("\n1. Enqueue");
      printf("\n2. Dequeue");
      printf("\n3. Display");
      printf("\n4. Exit");
      printf("\nEnter your choice: ");
      scanf("%d",&ch);
      switch(ch){
         case 1:
            printf("\nEnter the element to be inserted: ");
            scanf("%d",&val);
            enqueue(val);
            break;
         case 2:
            dequeue();
            break;
         case 3:
            display();
            break;
         case 4:
            printf("\nExit\n");
            break;
         default:
            printf("\nInvalid Choice\n");
      }
   }while(ch!=4);
   return 0;
}
```

Output on next page:

Output:

```
PS C:\Code\C\Code> cd "c:\Code\C\Code\" ; if ($?) { gcc QueueCircularArray.c -o QueueCircularArray } ; if ($?) { .\QueueCircularArray }

1. Enqueue
2. Dequeue
3. Display
4. Exit
Enter your choice: 1

Enter the element to be inserted: 5

Inserted element is: 5
1. Enqueue
2. Dequeue
3. Display
4. Exit
Enter your choice: 1

Enter the element to be inserted: 15

Inserted element is: 15
1. Enqueue
2. Dequeue
3. Display
4. Exit
Enter your choice: 1

Enter the element to be inserted: 25

Inserted element is: 25
1. Enqueue
2. Dequeue
3. Display
4. Exit
Enter your choice: 2

Deleted element is: 5
1. Enqueue
2. Dequeue
3. Display
4. Exit
Enter your choice: 3

Queue elements are:
15      25
1. Enqueue
2. Dequeue
3. Display
4. Exit
Enter your choice: 4

Exit
PS C:\Code\C\Code>
```

3. Linear Queue using Linked List

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

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

struct node *front = NULL;
struct node *rear = NULL;

void enqueue(int x){
   struct node *newNode = (struct node*)malloc(sizeof(struct node));
   newNode->data = x;
```

```
   newNode->next = NULL;
   if(rear == NULL){
      front = newNode;
      rear = newNode;
    }
   else{
      rear->next = newNode;
      rear = newNode;
    }
   printf("\nInserted element is: %d",x);
}

void dequeue(){
   if(front == NULL){
      printf("\nQueue is empty\n");
    }
   else{
      struct node *temp = front;
      printf("\nDeleted element is: %d",temp->data);
      front = front->next;
      free(temp);
   }
}

void display(){
   struct node *temp = front;
   if(front == NULL){
      printf("\nQueue is empty\n");
    }
   else{
        printf("\nQueue elements are:\n");
        while(temp != NULL){
            printf("%d ->",temp->data);
            temp = temp->next;
        }
    }
}

int main(){
   int ch, val;
   do{
      printf("\n1. Enqueue");
      printf("\n2. Dequeue");
      printf("\n3. Display");
      printf("\n4. Exit");
      printf("\nEnter your choice: ");
      scanf("%d",&ch);
      switch(ch){
```

```c
        case 1:
            printf("\nEnter the element to be inserted: ");
            scanf("%d",&val);
            enqueue(val);
            break;
        case 2:
            dequeue();
            break;
        case 3:
            display();
            break;
        case 4:
            printf("\nExit\n");
            break;
        default:
            printf("\nInvalid Choice\n");
    }
    }while(ch!=4);
   return 0;
}
```

Output:

```
PS C:\Code\C\Code> cd "c:\Code\C\Code\" ; if ($?) { gcc QueueLinearLL.c -o QueueLinearLL } ; if ($?) { .\QueueLinearLL }

1. Enqueue
2. Dequeue
3. Display
4. Exit
Enter your choice: 1

Enter the element to be inserted: 12

Inserted element is: 12
1. Enqueue
2. Dequeue
3. Display
4. Exit
Enter your choice: 1

Enter the element to be inserted: 41

Inserted element is: 41
1. Enqueue
2. Dequeue
3. Display
4. Exit
Enter your choice: 1

Enter the element to be inserted: 34

Inserted element is: 34
1. Enqueue
2. Dequeue
3. Display
4. Exit
Enter your choice: 3

Queue elements are:
12 ->41 ->34 ->
```

```
1. Enqueue
2. Dequeue
3. Display
4. Exit
Enter your choice: 2

Deleted element is: 12
1. Enqueue
2. Dequeue
3. Display
4. Exit
Enter your choice: 3

Queue elements are:
41 ->34 ->
1. Enqueue
2. Dequeue
3. Display
4. Exit
Enter your choice: 4

Exit
PS C:\Code\C\Code>
```

4. Circular Queue using Linked List

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

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

struct node *front = NULL;
struct node *rear = NULL;

void enqueue(int x){
   struct node *newNode = (struct node*)malloc(sizeof(struct node));
   newNode->data = x;
   newNode->next = NULL;
   if(rear == NULL){
      front = newNode;
      rear = newNode;
   }
   else{
      rear->next = newNode;
      rear = newNode;
   }
   rear->next = front;
   printf("\nInserted element is: %d",x);
}

void dequeue(){
   if(front == NULL){
      printf("\nQueue is empty\n");
   }
   else{
```

```
      struct node *temp = front;
      printf("\nDeleted element is: %d",temp->data);
      if(front == rear){
        front = NULL;
        rear = NULL;
      }
      else{
        front = front->next;
        rear->next = front;
      }
      free(temp);
   }
}

void display(){
   struct node *temp = front;
   if(front == NULL){
      printf("\nQueue is empty\n");
   }
   else{
      printf("\nQueue elements are:\n");
      do{
        printf("%d ->",temp->data);
        temp = temp->next;
      }while(temp != front);
   }
}

int main(){
   int ch, val;
   do{
      printf("\n1. Enqueue");
      printf("\n2. Dequeue");
      printf("\n3. Display");
      printf("\n4. Exit");
      printf("\nEnter your choice: ");
      scanf("%d",&ch);
      switch(ch){
        case 1:
            printf("\nEnter the element to be inserted: ");
            scanf("%d",&val);
            enqueue(val);
            break;
        case 2:
            dequeue();
            break;
        case 3:
            display();
```

```
            break;
        case 4:
            printf("\nExit\n");
            break;
        default:
            printf("\nInvalid Choice\n");
      }
   }while(ch!=4);
   return 0;
}
```

Output:

```
PS C:\Code\C\Code> cd "c:\Code\C\Code\" ; if ($?) { gcc QueueCircularLL.c -o QueueCircularLL } ; if ($?) { .\QueueCircularLL }

1. Enqueue
2. Dequeue
3. Display
4. Exit
Enter your choice: 1

Enter the element to be inserted: 53

Inserted element is: 53
1. Enqueue
2. Dequeue
3. Display
4. Exit
Enter your choice: 1

Enter the element to be inserted: 24

Inserted element is: 24
1. Enqueue
2. Dequeue
3. Display
4. Exit
Enter your choice: 1

Enter the element to be inserted: 89

Inserted element is: 89
1. Enqueue
2. Dequeue
3. Display
4. Exit
Enter your choice: 3

Queue elements are:
53 ->24 ->89 ->
1. Enqueue
2. Dequeue
3. Display
4. Exit
Enter your choice: 2

Deleted element is: 53
1. Enqueue
2. Dequeue
3. Display
4. Exit
Enter your choice: 3

Queue elements are:
24 ->89 ->
1. Enqueue
2. Dequeue
3. Display
4. Exit
Enter your choice: 4

Exit
PS C:\Code\C\Code>
```