Queue Implementation Using Arrays

Problem Statement: Implement a queue data structure using arrays. Your program should support the following queue operations:

1. Enqueue: Add an element to the rear of the queue.
2. Dequeue: Remove an element from the front of the queue.
3. Peek: Display the front element without removing it.
4. IsEmpty: Check if the queue is empty.
5. IsFull: Check if the queue is full (assume a fixed size).

Assignment Tasks:

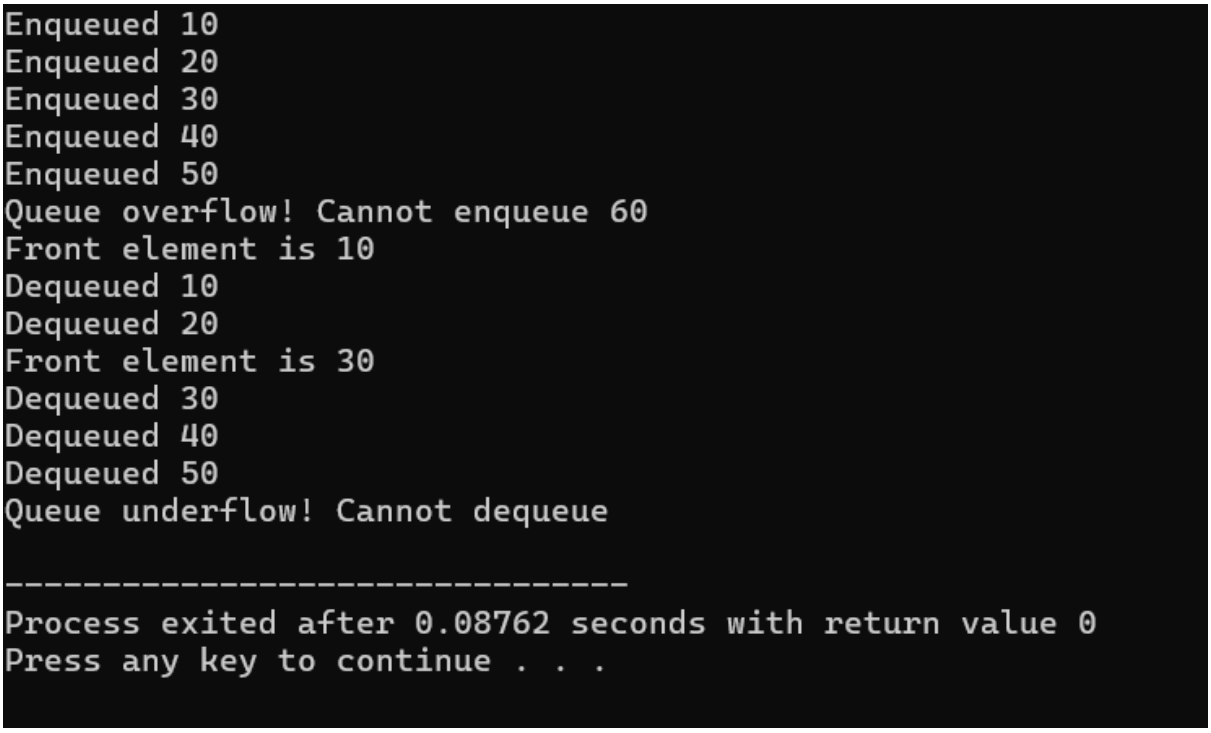
- Write a C program that defines a queue using arrays.
- Implement the queue operations mentioned above.
- Demonstrate queue overflow and underflow conditions.
- Write a main program to test all queue operations.

CODE :

```
#include <stdio.h>
#define MAX 5
int queue[MAX];
int front = -1, rear = -1;
int isFull() {          // Check the queue is full
   return rear == MAX - 1;
}
int isEmpty() {         // Check  the queue is empty
   return front == -1 || front > rear;
}
void enqueue(int value) {     // add element to the rear of the queue
   if (isFull()) {
      printf("Queue overflow! Cannot enqueue %d\n", value);
   } else {
      if (front == -1) front = 0;
      queue[++rear] = value;
      printf("Enqueued %d\n", value);
   }
}
void dequeue() {          //remove element from the front of the queue
   if (isEmpty()) {
      printf("Queue underflow! Cannot dequeue\n");
   } else {
      printf("Dequeued %d\n", queue[front++]);
      if (front > rear) front = rear = -1;
   }
}
void peek() {
   if (isEmpty()) {
      printf("Queue is empty! No element to peek\n");
   } else {
```

```
        printf("Front element is %d\n", queue[front]);
    }
}
int main() {
    enqueue(10);
    enqueue(20);
    enqueue(30);
    enqueue(40);
    enqueue(50);
    enqueue(60);
    peek();
    dequeue();
    dequeue();
    peek();
    dequeue();
    dequeue();
    dequeue();
    dequeue();
    return 0;
}
```

OUTPUT



Queue Implementation Using Linked Lists

Problem Statement: Implement a queue data structure using a linked list. Your program should support the following operations:

1. Enqueue: Add an element to the rear of the queue.
2. Dequeue: Remove an element from the front of the queue.
3. Peek: Display the front element without removing it.
4. IsEmpty: Check if the queue is empty.

Assignment Tasks:

• Write a C program that defines a queue using a singly linked list.
• Implement the queue operations mentioned above.
• Demonstrate queue operations using linked lists.
• Write a main program to test all queue operations

CODE:

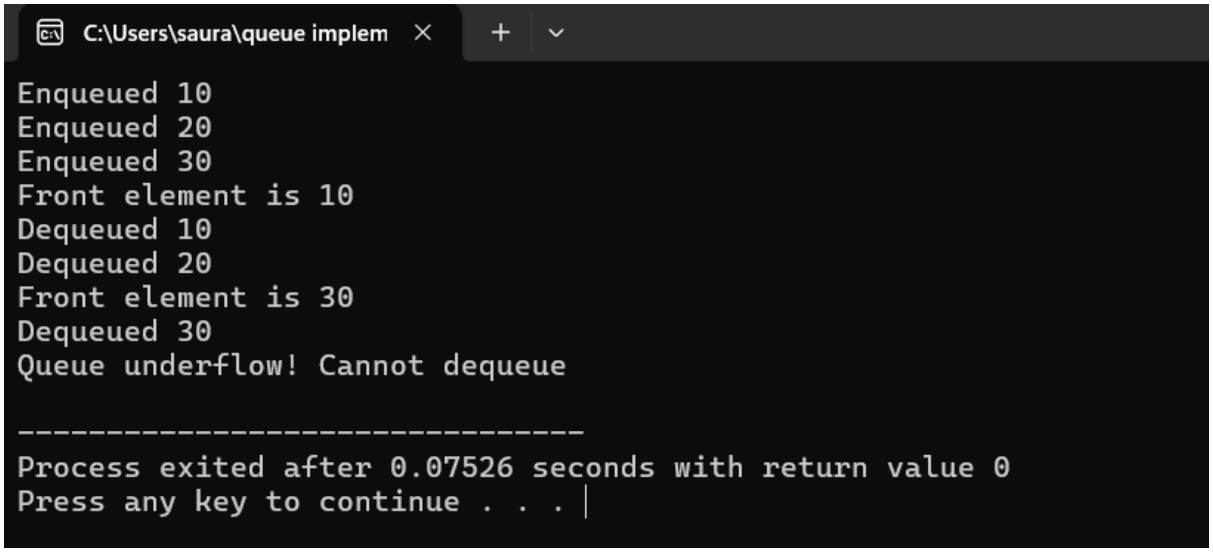
```
#include <stdio.h>
#include <stdlib.h>
struct Node {
   int data;
   struct Node* next;
};
struct Queue {
   struct Node* front;
   struct Node* rear;
};
void initialize(struct Queue* q) {
   q->front = q->rear = NULL;
}
int isEmpty(struct Queue* q) {
   return q->front == NULL;
}
void enqueue(struct Queue* q, int value) {
   struct Node* newNode = (struct Node*)malloc(sizeof(struct Node));
   if (!newNode) {
      printf("Memory allocation failed! Cannot enqueue %d\n", value);
      return;
   }
   newNode->data = value;
   newNode->next = NULL;

   if (isEmpty(q)) {
      q->front = q->rear = newNode;
   } else {
      q->rear->next = newNode;
      q->rear = newNode;
   }
   printf("Enqueued %d\n", value);
}
```

```
void dequeue(struct Queue* q) {
    if (isEmpty(q)) {
        printf("Queue underflow! Cannot dequeue\n");
        return;
    }
    struct Node* temp = q->front;
    printf("Dequeued %d\n", temp->data);
    q->front = q->front->next;
    if (q->front == NULL) {
        q->rear = NULL;
    }
    free(temp);
}
void peek(struct Queue* q) {
    if (isEmpty(q)) {
        printf("Queue is empty! No element to peek\n");
    } else {
        printf("Front element is %d\n", q->front->data);
    }
}
int main() {
    struct Queue q;
    initialize(&q);
    enqueue(&q, 10);
    enqueue(&q, 20);
    enqueue(&q, 30);
    peek(&q);
    dequeue(&q);
    dequeue(&q);
    peek(&q);
    dequeue(&q);
    dequeue(&q);

    return 0;
}
```

OUTPUT


C:\Users\saura\queue implem
Enqueued 10
Enqueued 20
Enqueued 30
Front element is 10
Dequeued 10
Dequeued 20
Front element is 30
Dequeued 30
Queue underflow! Cannot dequeue

Process exited after 0.07526 seconds with return value 0
Press any key to continue . . .