

# Linear Queue

```
#include <stdio.h>

#define SIZE 5 // Maximum size of queue

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

// Function to insert element
void enqueue(int value) {
    if (rear == SIZE - 1)
        printf("Queue is Full! (Overflow)\n");
    else {
        if (front == -1) front = 0;
        rear++;
        queue[rear] = value;
        printf("Inserted %d\n", value);
    }
}

// Function to delete element
void dequeue() {
    if (front == -1 || front > rear)
        printf("Queue is Empty! (Underflow)\n");
    else {
        printf("Deleted %d\n", queue[front]);
        front++;
    }
}
```

```

// Function to display queue elements

void display() {
    if (front == -1 || front > rear)
        printf("Queue is Empty!\n");
    else {
        printf("Queue elements: ");
        for (int i = front; i <= rear; i++)
            printf("%d ", queue[i]);
        printf("\n");
    }
}

// Main function

int main() {
    int choice, value;

    printf("---- Linear Queue Operations ----\n");
    while (1) {
        printf("\n1. Enqueue\n2. Dequeue\n3. Display\n4. Exit\n");
        printf("Enter your choice: ");
        scanf("%d", &choice);

        switch (choice) {
            case 1:
                printf("Enter value to insert: ");
                scanf("%d", &value);
                enqueue(value);
                break;
            case 2:
                dequeue();
                break;
        }
    }
}

```

```

case 3:
    display();
    break;
case 4:
    return 0;
default:
    printf("Invalid choice! Try again.\n");
}
}
}

```

## Circular queue

```

#include <stdio.h>
#define SIZE 5

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

// Function to insert element
void enqueue(int value) {
    if ((front == 0 && rear == SIZE - 1) || (rear + 1) % SIZE == front) {
        printf("Queue is Full! (Overflow)\n");
    } else {
        if (front == -1) // First element
            front = 0;
        rear = (rear + 1) % SIZE;
        queue[rear] = value;
        printf("Inserted %d\n", value);
    }
}

```

```
    }

}

// Function to delete element
void dequeue() {
    if (front == -1) {
        printf("Queue is Empty! (Underflow)\n");
    } else {
        printf("Deleted %d\n", queue[front]);
        if (front == rear) {
            // Only one element was present
            front = rear = -1;
        } else {
            front = (front + 1) % SIZE;
        }
    }
}
```

```
// Function to display elements
void display() {
    if (front == -1) {
        printf("Queue is Empty!\n");
    } else {
        int i = front;
        printf("Queue elements: ");
        while (1) {
            printf("%d ", queue[i]);
            if (i == rear)
                break;
            i = (i + 1) % SIZE;
        }
    }
}
```

```
    printf("\n");
}

}

// Main function
int main() {
    int choice, value;

    printf("---- Circular Queue Operations ----\n");
    while (1) {
        printf("\n1. Enqueue\n2. Dequeue\n3. Display\n4. Exit\n");
        printf("Enter your choice: ");
        scanf("%d", &choice);

        switch (choice) {
            case 1:
                printf("Enter value to insert: ");
                scanf("%d", &value);
                enqueue(value);
                break;
            case 2:
                dequeue();
                break;
            case 3:
                display();
                break;
            case 4:
                return 0;
            default:
                printf("Invalid choice! Try again.\n");
        }
    }
}
```

```
 }  
 }
```

# Double ended queue

```
#include <stdio.h>  
  
#define SIZE 5  
  
int deque[SIZE];  
int front = -1, rear = -1;  
  
// Insert at front  
void insertFront(int value) {  
    if ((front == 0 && rear == SIZE - 1) || (front == rear + 1)) {  
        printf("Deque is Full!\n");  
    } else if (front == -1) { // First element  
        front = rear = 0;  
        deque[front] = value;  
    } else if (front == 0) {  
        front = SIZE - 1;  
        deque[front] = value;  
    } else {  
        front--;  
        deque[front] = value;  
    }  
    printf("Inserted %d at front\n", value);  
}
```

```
// Insert at rear

void insertRear(int value) {

    if ((front == 0 && rear == SIZE - 1) || (front == rear + 1)) {
        printf("Deque is Full!\n");
    } else if (front == -1) { // First element
        front = rear = 0;
        deque[rear] = value;
    } else if (rear == SIZE - 1) {
        rear = 0;
        deque[rear] = value;
    } else {
        rear++;
        deque[rear] = value;
    }
    printf("Inserted %d at rear\n", value);
}
```

```
// Delete from front

void deleteFront() {

    if (front == -1) {
        printf("Deque is Empty!\n");
    } else if (front == rear) { // Only one element
        printf("Deleted %d from front\n", deque[front]);
        front = rear = -1;
    } else if (front == SIZE - 1) {
        printf("Deleted %d from front\n", deque[front]);
        front = 0;
    } else {
        printf("Deleted %d from front\n", deque[front]);
        front++;
    }
}
```

```
    }

}

// Delete from rear
void deleteRear() {
    if (front == -1) {
        printf("Deque is Empty!\n");
    } else if (front == rear) { // Only one element
        printf("Deleted %d from rear\n", deque[rear]);
        front = rear = -1;
    } else if (rear == 0) {
        printf("Deleted %d from rear\n", deque[rear]);
        rear = SIZE - 1;
    } else {
        printf("Deleted %d from rear\n", deque[rear]);
        rear--;
    }
}
```

```
// Display elements
void display() {
    if (front == -1) {
        printf("Deque is Empty!\n");
    } else {
        int i = front;
        printf("Deque elements: ");
        while (1) {
            printf("%d ", deque[i]);
            if (i == rear)
                break;
            i = (i + 1) % SIZE;
        }
    }
}
```

```

    }

    printf("\n");

}

}

// Main function

int main() {

    int choice, value;

    printf("---- Double Ended Queue (Deque) ----\n");

    while (1) {

        printf("\n1. Insert Front\n2. Insert Rear\n3. Delete Front\n4. Delete Rear\n5. Display\n6. Exit\n");

        printf("Enter your choice: ");

        scanf("%d", &choice);

        switch (choice) {

            case 1:

                printf("Enter value: ");

                scanf("%d", &value);

                insertFront(value);

                break;

            case 2:

                printf("Enter value: ");

                scanf("%d", &value);

                insertRear(value);

                break;

            case 3:

                deleteFront();

                break;

            case 4:

```

```
    deleteRear();
    break;

case 5:
    display();
    break;

case 6:
    return 0;

default:
    printf("Invalid choice! Try again.\n");

}

}

}
```