- By Rituraj
- 24BTDATA0033

~ DSA PROJECT ~

TOPIC : AIRPORT BAGGAGE MANAGEMENT SYSTEM

BRIEF SUMMARY :-

- This project simulates an Airport Baggage Handling System using Circular and Priority Queues in C. Circular Queue manages conveyor belt operations efficiently, while Priority Queue ensures timely processing of VIP and fragile baggage. The system mimics real-time baggage flow, optimizing sorting and handling to reduce delays and improve overall airport luggage management performance.

Problem Statement:

The project airport baggage by simulating real-time handling using Data Structures anaddresses the challenge of efficiently managing d Algorithms. It demonstrates how Circular Queues optimize conveyor belt usage and how Priority Queues ensure quick and organized processing of high-priority baggage, such as fragile or VIP items, enhancing speed and reliability in baggage systems.

Technologies Used:

- **Programming Language:** C
- **Data Structures:** Circular Queue, Priority Queue
- **Tools:** VS Code , Microsoft PowerPoint

WHAT IS QUEUE

- A queue is a linear data structure that follows the **First-In-First-Out (FIFO)** principle, where elements are added at the **rear** and removed from the **front**. It's commonly used in scenarios like task scheduling, printer spooling, and real-time data handling where order matters.

le

- There are two types of Queue used :-

Circular Queue

Priority Queue

CIRCULAR QUEUE :

A Circular Queue connects the last position back to the first, forming a loop. It efficiently utilizes memory by reusing freed space, making it ideal for fixed-size buffer management like in conveyor belt systems.

PRIORITY QUEUE :

A Priority Queue processes elements based on priority rather than arrival time. Higher-priority elements are served first, making it useful for scenarios like handling VIP or fragile baggage in airport systems where order of service matters.

ALGORITHM: AIRPORT BAGGAGE HANDLING SYSTEM

1. Start
2. Initialize Circular Queue for conveyor belt
3. Initialize Priority Queue for baggage based on priority
4. Input baggage details (ID, type, priority: Normal/Fragile/VIP)
5. If priority is high (VIP/Fragile),
    Insert baggage into Priority Queue
6. Else
    Insert baggage into Circular Queue
7. Process Priority Queue first (dequeue and display baggage)
8. Then Process Circular Queue (dequeue and display baggage)
9. Simulate baggage movement and updates
10. Repeat steps 4–9 until all baggage is processed
11. End

CODE :-

main.c > enqueue_circular(char [])

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

#define MAX_PRIORITY_QUEUE_SIZE 100

typedef struct {
    char baggage_id[50];
    int priority;
} PriorityNode;

PriorityNode priorityQueue[MAX_PRIORITY_QUEUE_SIZE];
int pqSize = 0;

#define CIRCULAR_QUEUE_SIZE 10

char circularQueue[CIRCULAR_QUEUE_SIZE][50];
int front = 0, rear = 0, count = 0;

void enqueue_priority(char baggage_id[], int priority);
char* dequeue_priority();
char* peek_priority();
void enqueue_circular(char baggage_id[]);
char* dequeue_circular();
char* peek_circular();
void baggage_system();
void swap(PriorityNode *a, PriorityNode *b);
void heapify_up(int index);
void heapify_down(int index);
```

main.c > enqueue_circular(char [])

```c
int main() {
    baggage_system();
    return 0;
}

void enqueue_priority(char baggage_id[], int priority) {
    if (pqSize == MAX_PRIORITY_QUEUE_SIZE) {
        printf("Priority queue is full!\n");
        return;
    }
    strcpy(priorityQueue[pqSize].baggage_id, baggage_id);
    priorityQueue[pqSize].priority = priority;
    heapify_up(pqSize);
    pqSize++;
    printf("Baggage %s added to priority queue with priority %d.\n", baggage_id, priority);
}

char* dequeue_priority() {
    if (pqSize == 0) {
        printf("Priority queue is empty!\n");
        return NULL;
    }
    static char baggage_id[50];
    strcpy(baggage_id, priorityQueue[0].baggage_id);
    priorityQueue[0] = priorityQueue[--pqSize];
    heapify_down(0);
    return baggage_id;
}
```

main.c > enqueue_circular(char [])

```c
char* peek_priority() {
    if (pqSize == 0) return NULL;
    return priorityQueue[0].baggage_id;
}

void heapify_up(int index) {
    while (index > 0) {
        int parent = (index - 1) / 2;
        if (priorityQueue[parent].priority >= priorityQueue[index].priority) break;
        swap(&priorityQueue[parent], &priorityQueue[index]);
        index = parent;
    }
}

void heapify_down(int index) {
    int largest = index;
    int left = 2 * index + 1;
    int right = 2 * index + 2;

    if (left < pqSize && priorityQueue[left].priority > priorityQueue[largest].priority)
        largest = left;
    if (right < pqSize && priorityQueue[right].priority > priorityQueue[largest].priority)
        largest = right;

    if (largest != index) {
        swap(&priorityQueue[index], &priorityQueue[largest]);
        heapify_down(largest);
    }
}
```

main.c main.exe

main.c > enqueue_circular(char [])

```c
void swap(PriorityNode *a, PriorityNode *b) {
    PriorityNode temp = *a;
    *a = *b;
    *b = temp;
}

void enqueue_circular(char baggage_id[]) {
    if (count == CIRCULAR_QUEUE_SIZE) {
        printf("Circular queue is full!\n");
        return;
    }
    strcpy(circularQueue[rear], baggage_id);
    rear = (rear + 1) % CIRCULAR_QUEUE_SIZE;
    count++;
    printf("Baggage %s added to circular queue.\n", baggage_id);
}
char* dequeue_circular() {
    if (count == 0) {
        printf("Circular queue is empty!\n");
        return NULL;
    }
    static char baggage_id[50];
    strcpy(baggage_id, circularQueue[front]);
    front = (front + 1) % CIRCULAR_QUEUE_SIZE;
    count--;
    return baggage_id;
}
```

main.c main.exe

main.c > enqueue_circular(char [])

```c
char* peek_circular() {
    if (count == 0) return NULL;
    return circularQueue[front];
}

void baggage_system() {
    while (1) {
        printf("\nAirport Baggage System\n");
        printf("1. Add baggage to priority queue\n");
        printf("2. Add baggage to circular queue\n");
        printf("3. Process baggage from priority queue\n");
        printf("4. Process baggage from circular queue\n");
        printf("5. View next baggage to be processed\n");
        printf("6. Exit\n");

        int choice;
        printf("Enter your choice: ");
        if (scanf("%d", &choice) != 1) {
            printf("Invalid input! Please enter a number between 1-6.\n");
            while (getchar() != '\n');
            continue;
        }
        char baggage_id[50];
        int priority;
        char* baggage;
        switch (choice) {
            case 1:
                printf("Enter baggage ID: ");
                scanf("%s", baggage_id);
                printf("Enter priority (1-10, 10 is highest priority): ");
```

```c
            printf("Enter priority (1-10, 10 is highest priority): ");
            if (scanf("%d", &priority) != 1 || priority < 1 || priority > 10) {
                printf("Invalid priority! Enter a number between 1 and 10.\n");
                while (getchar() != '\n');
                continue;
            }
            enqueue_priority(baggage_id, priority);
            break;
        case 2:
            printf("Enter baggage ID: ");
            scanf("%s", baggage_id);
            enqueue_circular(baggage_id);
            break;
        case 3:
            baggage = dequeue_priority();
            if (baggage) printf("Baggage %s processed from priority queue.\n", baggage);
            break;
        case 4:
            baggage = dequeue_circular();
            if (baggage) printf("Baggage %s processed from circular queue.\n", baggage);
            break;
        case 5:
            printf("Next baggage in priority queue: %s\n", peek_priority() ? peek_priority() : "None");
            printf("Next baggage in circular queue: %s\n", peek_circular() ? peek_circular() : "None");
            break;
        case 6:
            printf("Exiting the baggage system.\n");
            return;
        default:
            printf("Invalid choice. Please enter a number between 1-6.\n");
    }
  }
}
```

OUTPUT :-

```
Airport Baggage System
1. Add baggage to priority queue
2. Add baggage to circular queue
3. Process baggage from priority queue
4. Process baggage from circular queue
5. View next baggage to be processed
6. Exit
Enter your choice:
```

```
Airport Baggage System
1. Add baggage to priority queue
2. Add baggage to circular queue
3. Process baggage from priority queue
4. Process baggage from circular queue
5. View next baggage to be processed
6. Exit
Enter your choice: 1
Enter baggage ID: ABC123
Enter priority (1-10, 10 is highest priority): 10
Baggage ABC123 added to priority queue with priority 10.
```

```
Airport Baggage System
1. Add baggage to priority queue
2. Add baggage to circular queue
3. Process baggage from priority queue
4. Process baggage from circular queue
5. View next baggage to be processed
6. Exit
Enter your choice: 1
Enter baggage ID: DEF456
Enter priority (1-10, 10 is highest priority): 5
Baggage DEF456 added to priority queue with priority 5.
```

```
Airport Baggage System
1. Add baggage to priority queue
2. Add baggage to circular queue
3. Process baggage from priority queue
4. Process baggage from circular queue
5. View next baggage to be processed
6. Exit
Enter your choice: 2
Enter baggage ID: MNO789
Baggage MNO789 added to circular queue.
```

```
Airport Baggage System
1. Add baggage to priority queue
2. Add baggage to circular queue
3. Process baggage from priority queue
4. Process baggage from circular queue
5. View next baggage to be processed
6. Exit
Enter your choice: 2
Enter baggage ID: XYZ987
Baggage XYZ987 added to circular queue.
```

```
Airport Baggage System
1. Add baggage to priority queue
2. Add baggage to circular queue
3. Process baggage from priority queue
4. Process baggage from circular queue
5. View next baggage to be processed
6. Exit
Enter your choice: 3
Baggage ABC1233 processed from priority queue.
```

```
Airport Baggage System
1. Add baggage to priority queue
2. Add baggage to circular queue
3. Process baggage from priority queue
4. Process baggage from circular queue
5. View next baggage to be processed
6. Exit
Enter your choice: 3
Baggage DEF456 processed from priority queue.
```

```
Airport Baggage System
1. Add baggage to priority queue
2. Add baggage to circular queue
3. Process baggage from priority queue
4. Process baggage from circular queue
5. View next baggage to be processed
6. Exit
Enter your choice: 4
Baggage MNO789 processed from circular queue.

Airport Baggage System
1. Add baggage to priority queue
2. Add baggage to circular queue
3. Process baggage from priority queue
4. Process baggage from circular queue
5. View next baggage to be processed
6. Exit
Enter your choice: 4
Baggage XYZ987 processed from circular queue.
```

```
Airport Baggage System
1. Add baggage to priority queue
2. Add baggage to circular queue
3. Process baggage from priority queue
4. Process baggage from circular queue
5. View next baggage to be processed
6. Exit
Enter your choice: 5
Next baggage in priority queue: None
Next baggage in circular queue: None
```

Output END !!!!!


Thank You
Feeling gratitude and not expressing it is like wrapping a present and not giving it.