Air Traffic Scheduling System

Objective:
To enhance your programming and problem-solving skills, develop an **Air Traffic Scheduling System** in **C** that simulates runway allocation, flight scheduling, and air traffic management at a single airport. The system should dynamically adjust schedules based on real-time events, flight priorities, and constraints.

System Overview:

This system will manage incoming and outgoing flights at an airport. The goal is to allocate runways and schedule flights efficiently while minimizing delays and handling constraints like runway capacity and weather conditions.

Airline scheduling is a **complex process** that involves multiple steps. It requires collaboration between different stakeholders, including airlines, air traffic control, pilots, and ground staff. Below is the typical airline scheduling process:

Step 1: Flight Planning

- Airlines decide which flights will operate, considering demand, profitability, and aircraft availability.
- Each flight is assigned a **flight ID**, route, departure, and arrival time.
- Flights are categorized into **domestic, international, and emergency flights** based on their priority.

Step 2: Assigning Runways

- Each airport has a **limited number of runways** that must be used efficiently.
- Flights cannot take off or land simultaneously on the same runway.
- A **15-minute buffer time** is required between flights using the same runway.

Step 3: Crew Scheduling

- Crew members (pilots and flight attendants) are assigned based on their availability.
- **Duty time restrictions** ensure that crew members are not overworked.
- **Rest periods** must be provided before assigning a new flight to the crew.

Step 4: Managing Real-Time Events

- The system must handle unexpected delays caused by **weather conditions, technical issues, or emergencies**.

- Emergency landings are prioritized, and other flights may be rescheduled.
- Flight cancellations are updated in the system, and passengers are informed.

Step 5: Flight Execution and Monitoring

- Flights operate as per the assigned schedule unless an unexpected event occurs.
- **Runway utilization reports** are generated to analyze airport efficiency.
- Any delays or schedule changes are communicated to pilots and passengers.

Data set Creation

https://www.geeksforgeeks.org/how-to-design-database-for-flight-reservation-system/

Requirements:

Input:

1. **Flights**:
   - Flight ID (e.g., "AI101"), origin, destination.
   - Scheduled departure/arrival time (hh:mm).
   - Aircraft type (e.g., "Boeing 737", "Airbus A320").
   - Priority level (e.g., 1 for emergency, 2 for international, 3 for domestic).
2. **Runway Information**:
   Number of available runways at the airport (1–3 for simplicity).
3. **Crew Information :**
   - Pilot duty time restrictions
   - Crew rest requirements
   - Flight time limitations
   - Crew pairing constraints
4. **Real-Time Events**:
   - Weather delays (yes/no, time in minutes).
   - Emergency landing requests.
   - Flight cancellations.

Output:

1. **Scheduled Flights**:
   - Display the updated schedule for all flights (arrival/departure times, delays, runway assignments).
2. **Flight Conflicts**:

    - Notify if two flights have overlapping schedules for the same runway.
3. **Runway Utilization**:
    - Report on how each runway is being used over the simulation period.
4. **Delay Reports**:
    - Total delays for all flights, categorized by type (e.g., weather, priority conflicts).

Constraints:

1. **Runway Assignment**:
    - A flight cannot take off or land if another flight is using the same runway within a 15-minute buffer window.
2. **Flight Prioritization**:
    - Priority flights (e.g., emergencies) override other scheduled flights.
3. **Delays**:
    - Flights affected by weather delays must be rescheduled appropriately.
4. **Emergency Handling**:
    - Emergency flights must be assigned a runway immediately, overriding other schedules.
5. **Maximum Flights**:
    - The system should handle up to 50 flights at a time.

6. **Crew related:**

- Pilot duty time restrictions
- Crew rest requirements
- Flight time limitations
- Crew pairing constraints

Strategies to Implement:

1. **Data Structures**:
    - Use structures to represent flights, runways, and schedules
    - Use arrays to manage flights and runways
2. **Functions**:
    - **Add Flight**: Input flight details and assign it to a runway.
    - **Check Conflicts**: Detect overlapping schedules for the same runway.
    - **Reschedule Flight**: Adjust schedules dynamically when conflicts or delays occur.
    - **Handle Emergency**: Immediately assign a runway to priority flights.

- **Display Schedule**: Print the current schedule for all flights.

Bonus Challenges:

1. **Advanced Priority Management**:
   - Add a penalty for delayed high-priority flights in scheduling decisions.
2. **Simulation Mode**:
   - Automate flight arrivals and departures with random delays and cancellations.
3. **Interactive Mode**:
   - Allow users to dynamically add, modify, or cancel flights during execution.
4. **Graphical Output**:
   - Use ASCII art or basic visualization for schedules and runway utilization.

Deliverables:

1. Source Code: Self-documented C program with proper naming of variables and modular structure.
2. Test Cases: Include test cases for various scenarios, including ties, invalid inputs, and edge cases.
3. Report: A brief report explaining the following:
   - Approach and design choices.
   - Challenges faced and solutions implemented.
   - Key learnings and takeaways.
4. Demo: Provide a demonstration of the program, showcasing the key functionalities.