**PROJECT REPORT: ROVER MANAGEMENT SYSTEM**

**Objective:**

This project manages rovers and their parts using different data structures like Stack, Queue, Linked List, and Circular Linked List.

**1. Stack for Parts (LIFO - Last In First Out)**

* **Purpose:** Manage repair parts.
* **Operations:**
  + **push():** Add a part to the top.
  + **pop():** Remove the top part.
* **Example:**
  + Parts like Drill, Wheel, Camera, etc., are added and removed in reverse order.

**2. Queue-like Storage for Rovers (FIFO - First In First Out with Overwrite)**

* **Purpose:** Store rovers ready for deployment.
* **Operations:**
  + **insert\_rover():** Add a rover.
  + If full, remove the oldest rover.
* **Example:**
  + Rovers Rov1 to Rov7 are added. When full, oldest rovers are replaced.

**3. Singly Linked List for Faulty Rovers**

* **Purpose:** Track rovers that need repair.
* **Operations:**
  + **insert\_singly():** Add a faulty rover.
  + **delete\_singly():** Remove a rover once repaired.
* **Example:**
  + Rov6 and Rov3 are added. Rov3 is repaired and removed.

**4. Doubly Linked List for Repaired Rovers**

* **Purpose:** Track rovers after repair.
* **Operations:**
  + **insert\_doubly():** Add a repaired rover.
  + **traverse\_forward():** Move forward through the list.
  + **traverse\_backward():** Move backward.
* **Example:**
  + Rov3 is added to the repaired list.

**5. Circular Linked List for Upgraded Rovers**

* **Purpose:** Manage upgraded rovers.
* **Operations:**
  + **insert\_circular():** Add a new upgraded rover.
  + **traverse\_circular():** Display rovers in a loop.
* **Example:**
  + Rov1 and Rov4 are upgraded and shown in cycles.

**Conclusion:**

This project shows how different data structures can be used for efficient management of rovers and parts, helping in quick repairs, storage, and upgrades.