* PROBLEM STATEMENT

In today's fast-paced world, waiting in long hospital queues can be incredibly frustrating and time-consuming. Typically, hospital queues are managed manually by administrative staff, requiring patients to make an appointment, wait for their turn, and inquire about doctor availability. What makes this even worse is travelling a long distance, only to find out that the doctor is unavailable or on leave, making the trip unnecessary.

Developing a Hospital Management System enables the administration to efficiently manage patient data, doctor appointments, and hospital services. The system provides functionalities for both hospital staff (administrators, doctors) and patients.A Hospital Management System (HMS) can address multiple issues by allowing patients to book appointments from home and check the availability of their preferred doctor in advance. Doctors, in turn, can confirm or decline appointments, helping both parties. All the information is stored centrally by the system in files for easy access and updation. Bills can also be generated by the system to ensure ease of payment.

The goal is to create a menu driven console application with appropriate classes and methods, implementing object oriented principles like inheritance, polymorphism, and file handling.

* OBJECTIVE

The primary objective of the **Hospital Management System (HMS)** is to streamline and automate various administrative and clinical operations within a healthcare facility. By digitising patient data, appointment scheduling, and billing processes, HMS aims to:

1. **Improve Patient Experience**: Allow patients to book appointments, check doctor availability, and make payments online, reducing the need for long waits and unnecessary hospital visits.
2. **Enhance Doctor and Staff Efficiency**: Enable doctors to manage their appointments, confirm or decline bookings, and access patient records digitally, improving workflow and reducing administrative burden.
3. **Centralize Data Management**: Ensure all patient records, doctor information, and hospital services are stored in a centralised file, allowing quick retrieval and updating of information.
4. **Increase Operational Efficiency**: Automate tasks such as patient registration, appointment scheduling, and billing to reduce manual errors and administrative delays, ensuring smoother hospital operations.
5. **Enhance Communication**: Foster better communication between patients and doctors by providing timely updates on appointment status and medical records.

By achieving these objectives, the HMS contributes to improved healthcare services, better patient outcomes, and efficient hospital management.

* KEY FEATURES

1. Patient Management:

* Add a new patient (store details like patient ID, name, age, gender, ailment).
* View patient information.
* Update patient records (for example, updating treatment or discharge details).
* Delete patient records after discharge.
* Search patient by ID or name.

2. Doctor Management:

* Add, update, and delete doctor information (store details like doctor ID, name, specialty, and availability).
* View a list of all doctors.
* Assign a doctor to a patient.

3. Appointment Scheduling:

* Schedule appointments between patients and doctors based on availability.
* Cancel or reschedule appointments.
* View upcoming appointments for a specific patient or doctor.

4. Billing System:

* Generate and display the bill for a patient, including charges for treatments, tests, and hospital services.
* Payment status: paid or unpaid.

5. File Handling:

* Store all patient and doctor information in files to ensure persistence between program runs.
* Retrieve and update records from the file system.

6. Authentication:

* Admin login for managing hospital resources.
* Patient login to view personal information and upcoming appointments.

 Implementation:

* The system will have classes and objects to model patients, doctors, appointments, and bills.
* Use file I/O for saving and retrieving data.
* Focus on error handling for invalid inputs (e.g., duplicate patient IDs or scheduling conflicts).