

# Library Book Tracking and Late Fee Management

## Scenario

A library wants to automate tracking of issued books, check if members have borrowed essential reference books, and generate unique transaction codes for auditing. The system should also identify users who owe late fees.

## Requirements & Logic

### 1. Data Structures Setup

- **Dictionary (Issued\_Books):**

Keys = Member IDs (e.g., 'M-102 '),

Values = Lists of issued book titles.

```
{ 'M-102': ['Python Basics', 'AI Fundamentals'],  
'M-205': ['Data Science', 'OS Concepts'] }
```

- **Set (Reference\_Books):**

Contains titles that must be borrowed by every member at least once.

```
{ 'Python Basics', 'DBMS Principles' }
```

### 2. Filtering and Analysis (Loop/in/Update)

- Create a list `Late_Fee_Members` to store Member IDs of users with pending late fees.
- Use a `for` loop to go through each Member ID in `Issued_Books`.
- **Membership Check:**  
If 'Python Basics' is not in the issued list, print a warning that the member missed a required reference.
- **Condition Check:**  
If a member has borrowed **3 or more books**, assume a late fee might apply — append their ID to `Late_Fee_Members`.

### 3. String Processing and Formatting

- Convert all Member IDs (dictionary keys) into a tuple `Member_Tuple`.
- Create a transaction string `Audit_Code` by joining all IDs with a `#`.
- Slice the `Audit_Code` from index 2 to 9.
- Convert to uppercase using `.upper()`.

- Use `.format()` to print:
  - Total members with late fees
  - Final uppercase `Audit_Code`

### **Expected Output Example**

```
Member M-102 missed required reference book.  
Total Members with Late Fees: 1  
Final Audit Code: 102#M-2  
Late_Fee_Members: ['M-205']
```