# 30-Day C++ OOPs + STL Practice Ladder

### Day 1: Create a Class and Print Attributes (OOPs - Class, Access Specifiers)

Create a class `Person` with two public attributes: `name` (string) and `age` (int). Create an object and print the attributes.

**Output Example:**

Name: Alice

Age: 25

### Day 2: Student Class with Average Calculator (OOPs - Constructor, Member Functions)

Create a class `Student` with marks in 3 subjects as private data. Use a constructor to initialize them. Write a member function to calculate and print the average marks.

**Output Example:**

Average Marks: 85.33

### Day 3: Implement a Pair Class Template (Templates, OOPs)

Write a template class `MyPair<T1, T2>` that stores two values. Add a function to print both values.

**Output Example:**

First: John

Second: 42

### Day 4: Custom Comparator for Sorting Pairs (STL - pair, vector, sort)

Given a vector of pairs of (int, int), sort them by second value descending. Use a custom comparator.

**Output Example:**

```
(2, 90) (1, 85) (3, 80)
```

### Day 5: Bank Account Class with Static Members (Static members, Encapsulation)

Create a `BankAccount` class with a static variable `totalAccounts`. Increment it for each new account. Show number of accounts created.

**Output Example:**

```
Accounts Created: 3
```

### Day 6: Library System - Issue & Return Book (OOPs - Object Management)

Create a `Library` class where you can issue and return books. Track how many books are issued.

**Output Example:**

```
Books issued: 2
Books available: 3
```

### Day 7: Build a Mini Contact Manager (OOPs - Class + vector<string>)

Create a `Contact` class and use a vector to store multiple contacts. Add a function to display all.

**Output Example:**

```
John
Alice
Bob
```

### Day 8: Sort Students by Marks using Vector of Objects (STL - sort, custom comparator)

Create a `Student` class with name and marks. Store multiple objects in a vector and sort by marks.

**Output Example:**

```
Alice: 95

Bob: 90

John: 88
```

### Day 9: Basic Vector Operations (STL - vector)

Perform vector operations: push, pop, insert at index, and print vector elements.

**Output Example:**

```
Vector: 10 20 15 30
```

### Day 10: Word Frequency Counter (STL - map, string manipulation)

Take a sentence and use a map to count frequency of each word.

**Output Example:**

```
Hello: 2

World: 1
```

### Day 11: Implement Stack using STL (STL - stack, class wrapper)

Wrap STL `stack` inside a class `MyStack`. Add push, pop, and top functions.

**Output Example:**

```
Top: 5
```

### Day 12: Vehicle Inheritance Hierarchy (Single Inheritance, Base/Derived classes)

Base class: Vehicle. Derived class: Car. Print features from both.

**Output Example:**

```
Vehicle: 4 wheels

Car: AC included
```

**Day 13: Shape Class with Area Calculation (Polymorphism)**

Create base class `Shape` with virtual `area()` and derived classes `Circle`, `Square`.

**Output Example:**

Area of Circle: 78.5

Area of Square: 25

**Day 14: Bank Account - Savings & Current (Function Overriding)**

Base: `Account` with `displayBalance()`. Override it in `Savings` and `Current`.

**Output Example:**

Savings Balance: 1000

Current Balance: 2000

**Day 15: Protected Data in Inheritance (Protected Access + Inheritance)**

Base class has `protected: int id`. Derived class sets and prints it.

**Output Example:**

ID: 101

**Day 16: Task Scheduler with Priority Queue (STL - priority_queue, pair)**

Store tasks with priority using priority_queue. Print tasks in priority order.

**Output Example:**

Task3 (Priority 5)

Task1 (Priority 3)

**Day 17: Remove Duplicates using Set (STL - set, vector)**

Given a vector with duplicates, remove them using set.

**Output Example:**

Unique elements: 1 2 3 4

### Day 18: Deque Operations - Palindrome Checker (STL - deque)

Use deque to check if a string is a palindrome.

**Output Example:**

Is Palindrome: Yes

### Day 19: Custom String Class with Operator Overloading (OOPs - Operator Overload)

Overload `+` to concatenate strings using a custom class.

**Output Example:**

HelloWorld

### Day 20: Queue using Two Stacks (STL - stack, logic)

Implement queue functions `enqueue`, `dequeue` using two stacks.

**Output Example:**

Front: 10

### Day 21: Movie Booking System Design (OOPs - Composition)

Design classes: Movie, Show, Booking. Create and book a show.

**Output Example:**

Booked Show: Avengers at 7PM

### Day 22: Group Anagrams (STL - unordered_map)

Given words, group anagrams using unordered_map.

**Output Example:**

```
[aet]: eat, tea
[ant]: tan, nat
```

### Day 23: Implement a Vector Template (OOPs + Templates)

Write a simple template class `MyVector<T>` with push and print.

**Output Example:**

```
1 2 3 4
```

### Day 24: Logger System Design (OOPs + map)

Class `Logger` that logs message with timestamp. Ignore if repeated within 10 secs.

**Output Example:**

```
Logged: Hello
Ignored: Hello
```

### Day 25: Abstract Base Class: Animal Sounds (Abstract Class, Virtual Functions)

Base class `Animal` with pure virtual `sound()`. Derived classes implement it.

**Output Example:**

```
Dog: Bark
Cat: Meow
```

### Day 26: Online Shopping Cart Design (OOPs - Class, STL)

Design classes: Product, Cart. Add/remove items and show total price.

**Output Example:**

```
Total: $320
```

### Day 27: Mini File System (Inheritance, Tree Structure)

Create Folder/File hierarchy. Add and list contents.

**Output Example:**

```
Root -> Folder1 -> FileA.txt
```

### Day 28: Custom Hash Function in Unordered Map (Advanced STL)

Use unordered_map with a custom hash function for a custom class.

**Output Example:**

```
Element found: True
```

### Day 29: LRU Cache using List and Map (STL - list, map)

Implement LRU cache with get/put using list+map.

**Output Example:**

```
Accessed: 3

Evicted: 1
```

### Day 30: Parking Lot System (OOPs + Inheritance + STL)

Design classes: Vehicle, ParkingLot. Park/unpark vehicles.

**Output Example:**

```
Car Parked: A1

Slot Freed: A1
```