🔥 C++ PRACTICE QUESTIONS — **CONSTRUCTOR ONLY**

🧠 QUESTION 1 — SignalDecoder (State Decoding at Construction)

Problem Statement

Design a class `SignalDecoder` that determines system status **only during object construction**.

Requirements

1. The class must contain:
    - A private static integer `signalCode` (initial value: `100`)
2. The constructor must:
    - Evaluate the current value of `signalCode`
    - Print output based on the following rules:
        - Divisible by `2` → `Stable Signal`
        - Divisible by `5` → `Warning Signal`
        - Divisible by both `2` and `5` → `Critical Signal`
        - Otherwise → `Unknown Signal`
3. After evaluation:
    - Increment `signalCode` by `10`
4. Each new object must work with the **updated** `signalCode`.

Constraints

- All logic must be inside the constructor
- No helper methods
- No logic in `main()` except object creation

Sample Input

```
SignalDecoder a;
SignalDecoder b;
SignalDecoder c;
```

Sample Output

```
Critical Signal
Critical Signal
Critical Signal
```

🧠 QUESTION 2 — LaunchCrew (One-Time Role Assignment)

Problem Statement

Create a class `LaunchCrew` that assigns roles **strictly during construction**.

Requirements

1. The class must contain:
    - A private static integer `roleIndex` (initial value: `1`)
2. The constructor must:
    - Assign roles in the following order:
        - Object 1 → `Commander`
        - Object 2 → `Navigator`
        - Object 3 → `Engineer`
    - Any object created after the third must print:

      ```
      No role available
      ```
3. After each constructor call:
    - Increment `roleIndex`

Constraints

- No arrays
- No loops
- No switch-case
- No functions other than constructor

Sample Input

```
LaunchCrew c1;
LaunchCrew c2;
LaunchCrew c3;
LaunchCrew c4;
```

Sample Output

```
Commander
Navigator
Engineer
No role available
```

🧠 QUESTION 3 — TimeGate (Timeline-Based Validation)

Problem Statement

Design a class `TimeGate` that validates access based on a simulated timeline using only constructors.

Requirements

1. The class must contain:
    - A private static integer `timeSlot` (initial value: `0`)
2. The constructor must:
    - Allow access only if `timeSlot` is even
    - Print:
        - `Entry allowed at time X` when even
        - `Entry blocked at time X` when odd
3. After every constructor call:
    - Increment `timeSlot` by `1`

Constraints

- No input parameters
- No external time source
- No member functions except constructor

Sample Input

```
TimeGate t1;
TimeGate t2;
TimeGate t3;
TimeGate t4;
```

Sample Output

```
Entry allowed at time 0
Entry blocked at time 1
Entry allowed at time 2
Entry blocked at time 3
```
