GROUP - A

Answer To the Question Number: 1

The Template Method Pattern defines the skeleton of an algorithm in a method, deferring some steps to subclasses. Template Method lets subclasses redefine certain steps of an algorithm without changing the algorithm’s structure.

**Motivation:**

1. Minimizes duplication reusing code among subclasses
2. Only one class is concentrated about the algorithm & code changes only need to made there.
3. It provides a framework where a new subclass can be added only implementing a couple of methods.

The classes and objects participating in this pattern include:

1. **Abstract Class:** It implements a template method defining the skeleton of an algorithm. The template method calls primitive operations as well as operations defined in Abstract Class or those of other objects. It also defines abstract primitive operations that concrete subclasses define to implement steps of an algorithm.
2. **Concrete Class:** It implements the primitive operations to carry out subclass-specific steps of the algorithm.

Answer To the Question Number: 4

**Delegation:** Delegation is a way of making composition as powerful for reuse as inheritance. In delegation, two objects are involved in handling a request: a receiving object delegates operations to its delegate. This is analogous to subclasses deferring requests to parent classes.

Delegation is an extreme example of object composition. It shows that you can always replace inheritance with object composition as a mechanism for code reuse.

Several design patterns use delegation. The State, Strategy, and Visitor patterns depend on it.

Delegation has a disadvantage it shares with other techniques that make software more flexible through object composition, there are also run-time inefficiencies, but the human inefficiencies are more important in the long run.

Delegation is a good design choice only when it simplifies more than it complicates.