CMR ENGINEERING College

B.Tech. IV - I Examination December-2025

DESIGN PATTERNS (R22AI731PE)

Prepared by: DR. S. RASHEED UDDIN

PART-A

1a) LIST any two Categories of design Patterns?

A) a) Creational Patterns

b) Behavioural Patterns.

b) what are the benifits of using design Patterns?

A) Design Patterns benifits are

Improving code reusability, maintain-bility, Scalability and readability.

c) what are user operations in document-editor?

A) user operations in a document like text manipulation, formatting, structural additions, file management adv features like collaboration.

d) what is Spell Checking?

A) Spell checking is a software feature that identifies and corrects spelling mistakes in a text by comparing words against a digital dictionary.

e) List difference between Factory method and Abstract Factory.

A Factory method and Abstract Factory are both Creational design Patterns, but they differ in their scope and Complexity.

No. of Products:

⇒ Factory method creates Single Product.

⇒ Abstract Factory creates dependent Products.

f) Give a real world example of Singleton

A) Real world example of a Singleton Pattern is Global Logger or Configuration Manager in a Software Application.

g) Define Composite Objects?

A) A Composite object is a structure that combins multiple, or Simple objects to form a more complex structure to be treated as Single unified entity.

h) Provide an example of Decorator Pattern?

A Common example for decorator Pattern is a Coffee shop application where you can dynamically add topping like milk and sugar.

i) What is the Role of Visitor Pattern?

A. The Visitor Pattern role is to separate algorithms / operations from the object-structures they act upon, allowing you to add new behaviors to existing classes without modifying the code.

J) Define Strategy Pattern?

A. The Strategy Pattern is a behavioral design pattern that defines a family of interchangable algorithms. encapsulates each one in a separate class.

PART-B

2. What is a Catalog? Explain Catalog of design Pattern.

A. A Catalog of Design Patterns organizes reusable solutions to common software design problems. Categories mainly by their purpose. Creational (Obj creation), Structural (class / object composition) and Behavioral with some adding Concurrency for multi threaded task. helping developers find and Apply Solutions faster for more flexible maintainable code.

A Catalog of Design Pattern: is a structural collection of established solutions to recurring problems in software design, documented in a way that makes them easy to find and apply.

Think of it as cookbook for developers, Providing proven recipes for common scenarios. like how build complex objects.

which are mainly classified into Three Types

1) Creational Patterns
2) Structural Patterns
3) Behavioral Patterns.

3 What is a Design Pattern? How it supports to develop the application?

A. Design Pattern is a reusable blue print for solving common software design problems, offering proven, general solutions that developers can adapt, rather than direct code.

They support the application deployment by speeding up the process, improving code quality through establishing best practices and providing shared vocabulary for teams, leading to more robust and scalable applications

A How it Supports Application Development

1) Speeds up development.

2) Improve Code Quality.

3) Enhances Communication.

4) Promotes Consistency & Best Practices.

5) Facilitates Scalability.

6) Reduces Complexity.

4) How to Design a Document Editor? Explain.

A. Designing a Document Editor involves addressing key challenges, from defining the internal representation of the document to implementing user operations and supporting different system environments. A common approach by the Lexi case study.

The design process can be broken down into the following key areas.

1. Document Structure Representation

2) Formatting & Layout-

3) User Interface & Embellishment-

4) User Operations & Actions.

5) Supporting Different Systems & Analysis

6) Modern Considerations.

5. Describe how Patterns help solve Problems related to window systems and UI Support-

A Patterns are Crucial in Solving Problems related to windows Systems and UI Support by Providing reusable Proven Solutions to Common recurring issues. They Streamline troubleshooting, development and Support Process leading to faster resolutions and more consistent user experience.

1) Trouble shooting & Diagnostics.

2) UI design & development-

3) System Automation & management.
   i) Configuration mgmt.
   2) Log Analysis.

6 Describe the Builder Design Pattern?

A. The Builder design Pattern is a creational design Pattern that Separates the Construction of a Complex object from its representation, allowing the same Construction Process to create different types and representation of a Product. It enables the step-by-step Construction of an object, Providing a clear and readable way to build objects with numerous optional parameters or Complex configurations.

Key Components of Builder Design Patterns are:

1) Product
2) Concrete Builders
3) Director

7. Explain Abstract- factory design Pattern?

A. The Abstract Factory design Pattern is a Creational design Pattern that- Provides an interface for Creating families of related or dependent Objects without specifying their Concrete Classes. It is essentially a factory of afactories.

Key Concepts.

Abstract Factory

This is an interface or abstract Class that declares set of methods for Creating abstract Products.

Concrete Factories :

These are the implementation of the Abstract Factory interface. Each Concrete Factory is responsible for Creating Specific family of related Products.

Abstract Products:

These are interfaces or abstract classes that define the common interface for a type of Product with in the family.

Concrete Products:-

These are specific implementations of the abstract Products, Creating Concrete factories.

Client:

The Client Code interacts with abstract Factory & Abstract Products through their interfaces without needing to know the Concrete Classes being instantiated.

8. What is a Structural Pattern? Explain Composite design Patterns.

A. Structural Patterns organize classes/objects to simplify system structure, and the Composite Pattern, a key StructuralPattern, Composite objectinto tree like structures and treat individualobjects and their Compositions uniformly like files/ folders or menu items/ submenus.

Structural Patterns focuson how classes and objects are Composed to form largeer structures, Simplifying the relationship and responsibilities with in a System.

To Provide flexible, efficient ways to build robust objectstructures, often by reusing existence classes and objects, leading to cleaner and more maintainable Code.

The Composite Design Pattern

Core Idea: BuildTree structures where

individual objects (leaves) and groups of objects (Composite) share the same interface, allowing them to be treated as single units.

Key Components:

1. Component (Interface / Abstract Class)

   Declares common operations for both single and complex objects.
   It often provides default implementations for container operations which leaves can override.

2. Leaf Single Object:

   Represent the basic Non-container objects.

3. Composite (Container Object)

4. Client.

9. Discuss the structure and working of Adapter & Bridge Patterns with examples.

A. The Adapter Pattern involves three key components.

Target Interface : This is the interface that the Client expects to interact with.

Adaptee :- This is an existing class with an incompatible interface that need to be made compatible with the Target Interface.

Adapter :- This class implements the Target Interface and contains an interface instance of the adaptee

Working :-

The Adapter acts as a wrapper around the Adaptee converting the Client's request into a format that the adaptee can understand.

When the client calls a method on the adapter, the Adapter, in turn calls the corresponding method on the Adaptee, possibly with some data transformation. This allows the Client use the adaptee without being aware of it's incompatible interface.

Ex :- Consider a legacy OldLogger class with a logMessage (String message) method and a new System expects a logger interface with a write(String text) method. on the

10. Explain the Strategy Pattern and Compare it with Template method.

The Strategy Pattern defines a family of Algorithms encapsulate each one, make them interchangable; It allows the Client to choose an Algorithm at run time without modifying the Context object that uses the Algorithm.

Strategy Pattern & Template Pattern defines the skeleton of an algorithm in a base class deferring the implementation of some steps to Sub classes. It allows Sub classes to redefine certain steps of an Algorithm without changing the Algorithms overall structure.

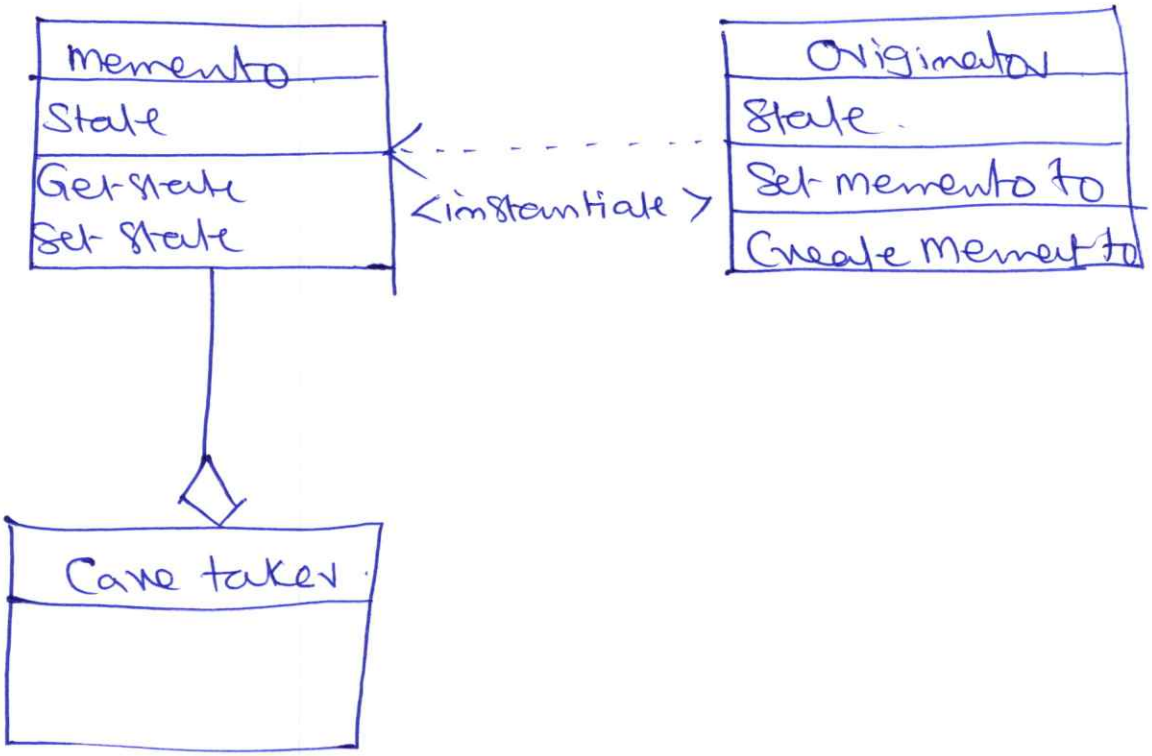
	Strategy Pattern	Template Pattern
1. Mechanism:	Composition delegating to a strategy object.	Inheritance overriding methods
2. Flexibility:	Runtime flexibility entire algorithm can be swapped.	Compile Time flexibility, only specific steps can be varied
3. Algorithm:	Encapsulates entire Algorithm	Defines Algorithm Skeleton.
4. Client-Role:	Client selects and provides the strategy object.	Client uses the base class & subclasses provide specific imple
5. Relationship:	Loose coupling b/w context & strategy	Tighter coupling b/w base class & subclass
6. Example:	Different sorting ~~Different~~ So Algorithms	Steps for making a beverage.

11. Explain the memento Pattern and discuss its applications.

A. The memento Pattern is a behavioral design Pattern that lets you Save and Restore an objects Previous State without breaking its encapsulation., acting like a Snapshot mechanism for undo/redo, history tracking and state roll back.

It involves three key components. the ~~organizer~~ Originator, the memento the object holding the Snapshot of the State. and the Caretaker which stores mementos, but it doesn't inspect them. Its main applications include text editor undo/redo, version control systems like Git.

Memento Pattern Diagram



—END—