

Software Design Concepts

IMPORTANT

(Explain with Examples)

- ✓ Abstraction
- ✓ Architecture & Design Patterns
- ✓ Modularity & Information Hiding
- ✓ Functional Independence
- ✓ Refinement & Refactoring
- ✓ Object-Oriented Design Concept



Ms. Vrushali Dhanokar

M.Tech CSE

dhanokarvrushali@gmail.com

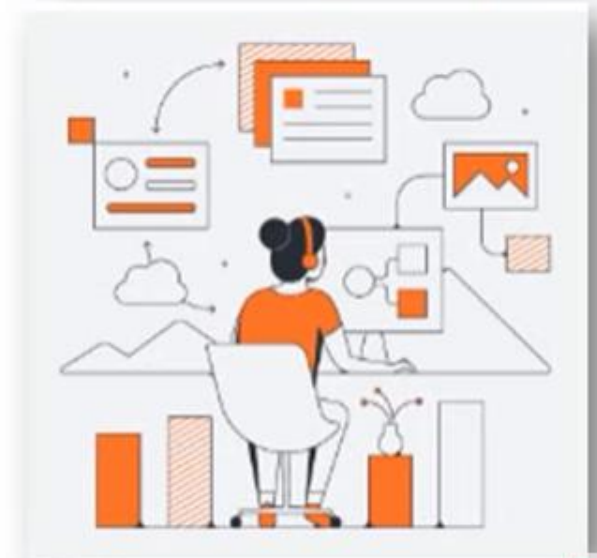
Contents

1. **Software Design Concepts**
2. **Important Questions**



About Software Design Concepts

- The software design concept simply means the idea or principle behind the design.
- It describes how you plan to solve the problem of designing software.
- It also shows the logic or thinking behind how you will design software.
- The software design concept for developing the right software provides a supporting and essential structure or model.



Software Design Concepts

1. Abstraction
2. Architecture 
3. Design Patterns
4. Modularity
5. Information Hiding
6. Functional Independence
7. Refinement
8. Refactoring
9. Object-Oriented Design Concept

1. Abstraction

- Abstraction is used to hide background details or unnecessary implementation about the data.
- So that users see only required information.

Type 1: Procedural Abstraction:

- There is collections of subprograms.
- One is hidden group another is visible group of functionalities. **Example:**



Private:

Fuel_machine()
Set_top_speed()
Develop_engine()

Public:

Turn_on() , Turn_off()
Accelerate(), Break()

Type 2: Data Abstraction:

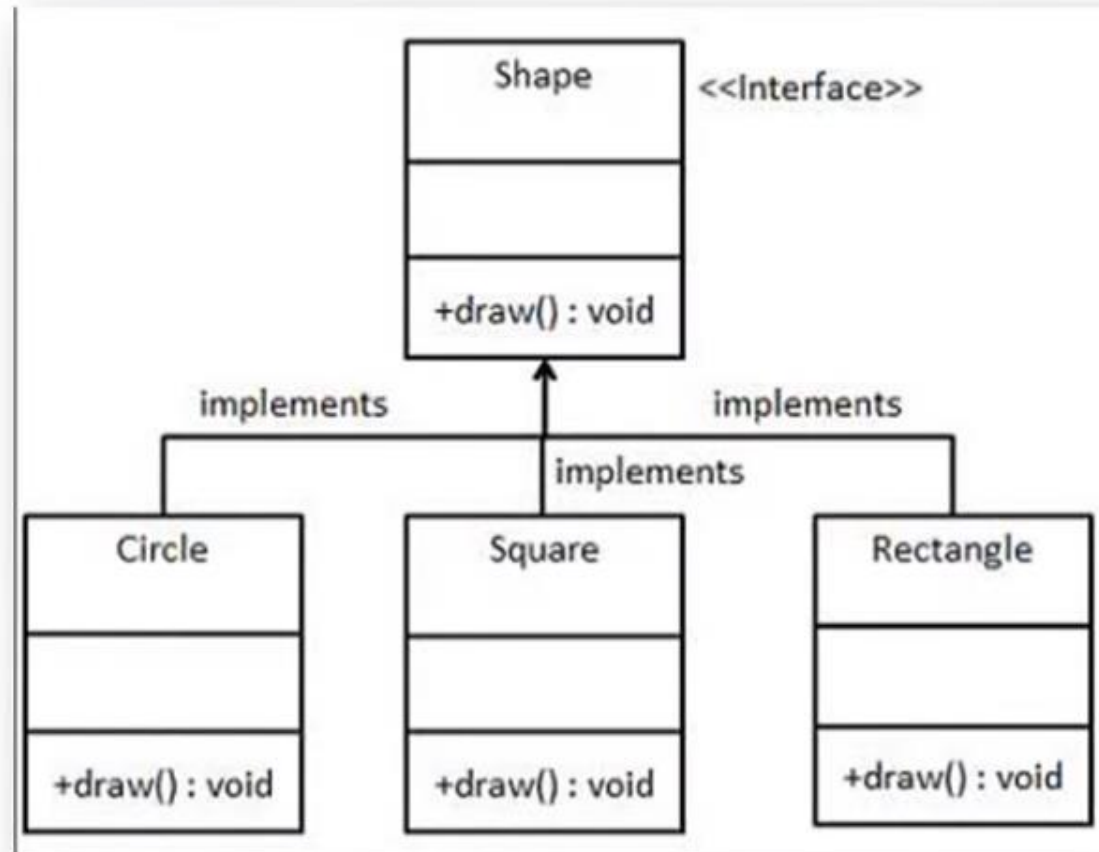
- Collections of data that describe data objects.
- Show representation data & hide manipulation data.
- **Examples:** Data Structure Programs directly used Push(), Pop(), Top() and Empty() methods.

2. Architecture

- The architecture is the structure of program modules where they interact with each other in a specialized way.
- **Structural Properties:** Architectural design represent different types of components, modules, objects & relationship between these.
- **Extra-Functional Properties:** How design architecture achieve requirements of Performance, Capacity, Reliability, Security, Adaptability & other System Characteristics.
- **Families of related systems:** The architectural design should draw repeatable patterns. They have ability to reuse repeatable blocks.

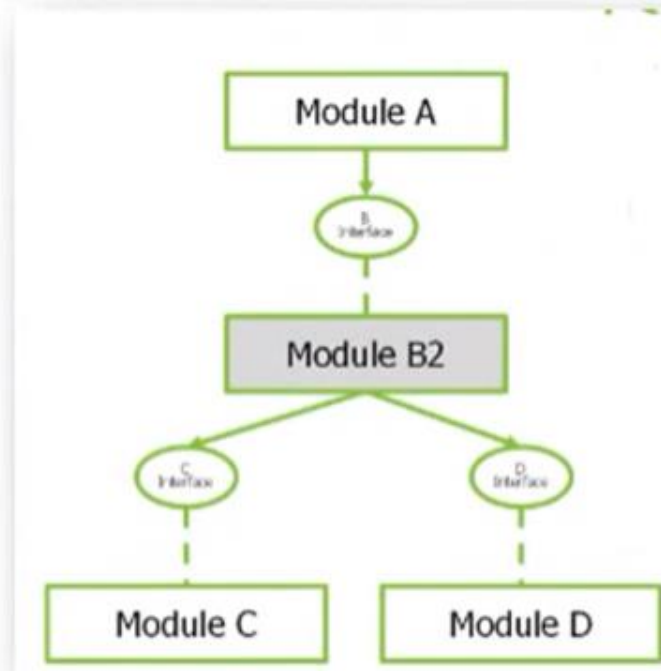
3. Design Patterns

- The pattern simply means a repeated form or design in which the same shape is repeated several times to form a pattern.
- **Example:**



4. Modularity

- Modularity simply means dividing the system or project into smaller parts to reduce the complexity of the system or project.
- After developing the modules, they are integrated together to meet the software requirements.
- Modularizing a design helps to effective development, accommodate changes easily, conduct testing, debugging efficiently and conduct maintenance work easily.



5. Information Hiding

- Modules should be specified and designed in such a way that the data structures and algorithm details of one module are not accessible to other modules.
- They pass only that much information to each other, which is required to accomplish the software functions.
- The way of hiding unnecessary details in modules is referred to as information hiding.

The image shows a web interface with a dark blue background. On the left, there is a white box titled 'Sign Up' with the subtitle 'It's free and only takes a minute'. It contains five input fields: 'First Name', 'Last Name', 'Email', 'Password', and 'Confirm Password'. Below these fields is a green 'Submit' button. At the bottom of the 'Sign Up' box, there is a line of text: 'By clicking the Sign Up button you agree to our [Terms and Condition](#) and [Policy Privacy](#)'. Below the entire form area, there is a link: 'Already have an account? [Login here](#)'. On the right, there is another white box titled 'Login'. It contains two input fields: 'Email' and 'Password'. Below these fields is a green 'Submit' button. Above the 'Login' box, there is a link: 'Not have an account? [Sign Up Here](#)'.

6. Functional Independence

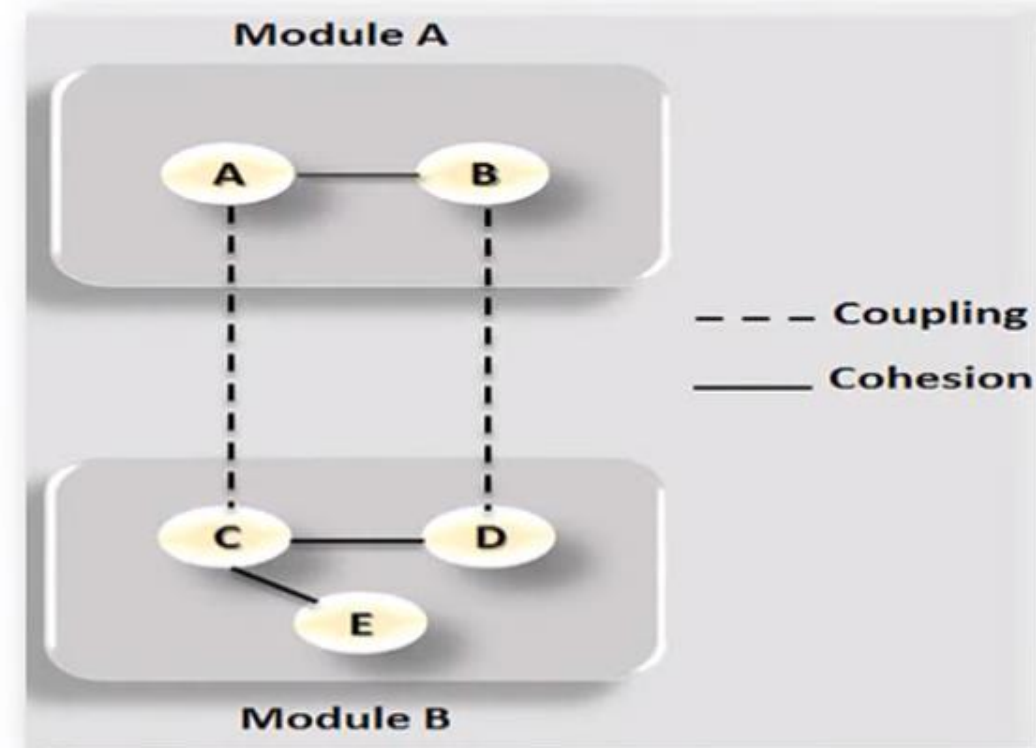
- The functional independence is the concept of separation and related to the concept of modularity, abstraction and information hiding.

Criteria 1: Coupling

- The degree in which module is “connected” to other module in the system.
- Low Coupling necessary in good software.

Criteria 2: Cohesion

- The degree in which module perform functions in inner module in the system.
- High Cohesion necessary in good software.



7. Refinement

- Refinement is a top-down design approach.
- It is a process of elaboration.
- A program is established for refining levels of procedural details.
- A hierarchy is established by decomposing a statement of function in a stepwise manner till the programming language statement are reached.

Example:

INPUT

Get number 1 (Integer)

Get number 2 (Integer)

PROCESS

OUTPUT

INPUT

Get number 1 (Integer)

Get number 2 (Integer)

While (Invalid Number)

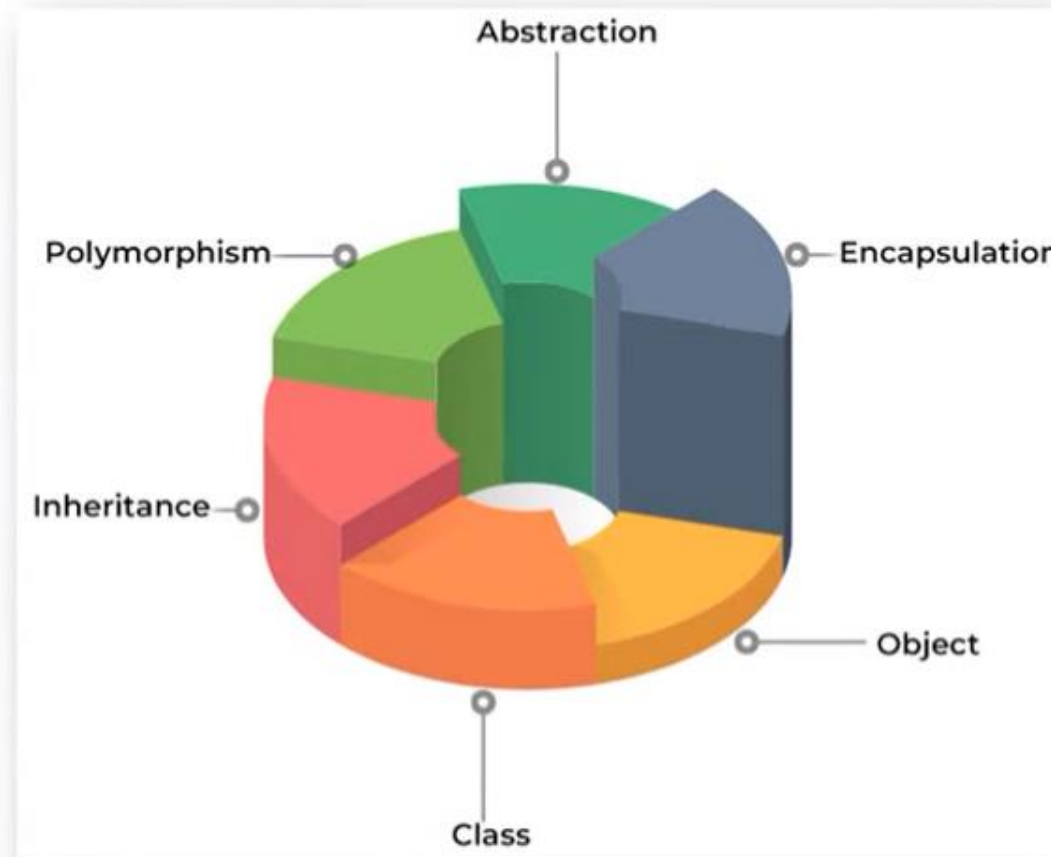
EXIT

8. Refactoring

- Refactoring is the process of changing the internal software system in a way that it does not change the external behavior of the code still improves its internal structure.
- When software is refactored the existing design is examined for redundancy, unused design elements, unnecessary design algorithms, poorly constructed data, inappropriate data structure or any other design failure that can be corrected for better design.

9. Object Oriented Design Concepts

- Object Oriented is a popular design approach for analyzing and designing an application.
- Advantage is that faster, low cost development and creates a high quality software.



Important Questions

As per previous year question papers

1. Explain different design concepts. 8M
2. What is the relationship between modularity & functional independence. 7M
3. What do you mean by term cohesion & coupling in the context of software design? How are the concepts are useful for good design system. 7M



Thank You...!



LIKE



COMMENT



SHARE



SUBSCRIBE



THANKS FOR WATCHING