Creational Design Patterns:

Purpose: Focuses on the process of object creation.
Deals with: How objects are instantiated and composed.

Examples:
Singleton: Ensures a class has only one instance and provides a global point of access to it.
Factory Method: Defines an interface for creating an object, but leaves the choice of its type to the subclasses.
Abstract Factory: Provides an interface for creating families of related or dependent objects without specifying their concrete classes.
Builder: Separates the construction of a complex object from its representation.
Prototype: Creates new objects by copying an existing object.

Structural Design Patterns:

Purpose: Focuses on the composition of classes or objects.
Deals with: How classes and objects are composed to form larger structures.

Examples:
Adapter: Allows the interface of an existing class to be used as another interface.
Decorator: Attaches additional responsibilities to an object dynamically.
Proxy: Provides a surrogate or placeholder for another object to control access to it.
Composite: Composes objects into tree structures to represent part-whole hierarchies.
Facade: Provides a simplified interface to a set of interfaces in a subsystem.

Behavioural Design Patterns:

Purpose: Focuses on the interaction and communication between objects.
Deals with: How objects collaborate and operate together.

Examples:
Observer: Defines a one-to-many dependency between objects so that when one object changes state, all its dependents are notified and updated automatically.
Strategy: Defines a family of algorithms, encapsulates each one, and makes them interchangeable.
Command: Encapsulates a request as an object, allowing for parameterization, queuing, and logging of requests.
Chain of Responsibility: Passes the request along a chain of handlers.
State: Allows an object to alter its behaviour when its internal state changes.
Iterator: Provides a way to access the elements of an aggregate object sequentially.
Memento: Captures and externalizes an object's internal state for later restoration.
Visitor: Represents an operation to be performed on the elements of an object structure.

https://canvas.ubc.ca/courses/133331/files/31812465?module_item_id=6647308
https://sourcemaking.com/design_patterns

10 SOFTWARE
DESIGN
PATTERNS