**Design Patterns- split to 3 groups**

|  |  |  |
| --- | --- | --- |
| **Creational Patterns** | **Structural Pattern** | **Behaviour pattern** |
| These are approaches that deal with the **creation of object**  **🡪 Provide Instantiation mechanisms**  🡪 Instead of instantiating all objects very specifically and very explicitly, this design pattern give us more **flexibility in how the objects are actually created** | This approach dealt more with **how classes are actually designed**,  🡪 This deals with relationship between entities, making it easier for these entities to work together  🡪 How things like **inheritance and composition and aggregation can be used** to provide extra functionality | Most of these design patterns are specifically concerned with **communication between objects** as the program is running. |
| * Abstract Factory * Builder * Factory Method * Prototype * Singleton | * Adapter * Bridge * Composite * Decorator * Façade * Flyweight * Proxy | * Chain of responsibility * Command * Interpreter * Iterator * Mediator * Memento * Observer * State * Strategy * Template method * Visitor |