1. Interface bao các class con và là đại diện cho các class con. Khi khởi tạo dùng interface:

Iterface a = new ImplementClass();

### ****Nhóm Creational (nhóm khởi tạo)****

**Singleton**:

-Trong class tạo 1 static instance (Có thể dùng static final khi tạo)

-private contructor để không cho phép tạo

-Hàm public static getInstance

**Factory Method**:

Gồm 2 phần:

* Factory: Class

Là một Class chứa hàm tạo class theo điều kiện (switch… case, if…else, Set).

Hàm tạo này áp dụng tính chất 1 => kiểu trả về là Interface

* Product: Interface

**Abstract Fatory**:

* Factory: Interface
* Product: Interface

**Builder**:

Class chứa class product. Chia nhỏ product ra thành nhiều phần và tạo các hàm tạo. Dùng thay contructor, tránh phải tạo nhiều contructor.

**Prototype**: Clone

### ****Nhóm Structural (nhóm cấu trúc)****

**Adapter**: 1 Interface bao 1 class

Interface

Class

**Facade**: 1 Interface bao nhiều class

Interface

Class

Class

Class

**Bridge**:

Abstract - Interface

Trong Abstract có thuộc tính “protect Interface instance” (Implement của interface này chứa interface khác),

Tạo contructor dùng super để truy cập tới contructor của abstract

Interface – interface2

Implement của interface này chứa interface khác

Tạo contructor truyền vào interface2

**Composite(Kết hợp)**

Gồm 3 thành phần:

- Component: Interface

- Leaf: implement interface (thằng con, thằng bé nhất)

- Composite: implement interface bao gồm 1 list<Component>, addLeaf(), removeLeaf() và các phương thức xử lý (dùng for in trong các hàm này để thao tác với tất cả các thằng con)

**Proxy**

Tạo một interface cho proxy và realObject.

Trong proxy có chứa realObject

K thao tác trực tiếp với realOject mà thao tác thông qua proxy

### ****Nhóm Behavioral (nhóm hành vi/ tương tác)****

**Strategy**:

Dùng một class bao lại một interface. Sử dụng hàm của interface

**Observer**:

Gồm 2 thành phần:

- Publisher(bao gồm list các subcribers), có hàm notify() sẽ duyệt qua tất cả các con và gọi phương thức receive() của chúng

-subscriber(Interface có phương thức receive)

**Command**

Tách đối tượng và lệnh thực hiện hiện đối tượng ra (Không trực tiếp gọi hàm của đối tượng đó mà gọi qua một lớp khác để tang tính bảo mật và quản lý được khi nào thì gọi)

Gồm:-

- Đối tượng

- ICommand (interface) có hàm excecute (Các implementer sẽ dùng hàm này để gọi đến hàm của đối tượng).

- Invoker (class để gọi đến execute để thực hiện lệnh)

**Interator (người thực hiện vòng lặp)**

Gồm: 2 interface (IInterator-người lặp và IAggregate – list đối tượng)

IInterator gồm 2 phương thức là hasNext() và Object next(). Implementor chứa 1 aggregate và int index

IAggregate gồm IInterator createInterator(). Implementer chứa list cần truy cập

**Specification** (dùng để tách điều kiện kiểm tra ra khỏi logic xử lý)

Gồm 1 interface có hàm boolean isSatisfiedBy()

Các implementer gồm: - các composite điều kiện vd: and, or, not

-Các class điều kiện

**Pooling:** Tạo một list để lưu trữ các đối tượng đã tạo để tái sử dụng. Kiểu một dạng queue: dùng thì lấy ra, xong thì đẩy lại