Simple Factory
SIMPLE FACTORY allows interfaces for creating objects without exposing the object creation logic to the client.
```java
SimpleFactory

+ CreateProduct(enum type)

switch(type):
case A:
    return new ProductA();
case B:
    return new ProductB();

Client

ProductA

ProductB
```
THE BAD

- Breaks open/closed principle
- References implementation directly
FACTORY METHOD
FACTORY METHOD defines an interface for creating an object, but let subclasses decide which class to instantiate.

It lets a class defer instantiation to subclasses.
STRUCTURE

```
SomeBehavior() {
    product = MakeProductB();
    ...
}
```

```
ConcreteCreator
+ MakeProductA()
+ MakeProductB()
+ MakeProductB()

return new ConcreteProductB()
```
Abstract Factory
ABSTRACT FACTORY provides an interface for creating families of related or dependent objects without specifying their concrete classes.
STRUCTURE
ABSTRACT FACTORY does not specify how the objects should be created. It only enforces that objects are created in families.

To create the objects it is usually paired with either the factory method or the prototype pattern.
- Read the **Factory** chapter in CGLearn
- Do the **Factory** task in CGLearn **IN TEAMS**