어떤 클래스를 만들지 결정한다

-> 그 클래스가 가져야할 상태, 행동이 무엇인지 생각한다.

-> 꼭 필요한 상태와 필요하지 않은 상태를 나누고 꼭 필요한 상태의 경우 생성자로 초기화한다.

-> 이때 행동은 다른 객체와의 관계를 생각해서 만들어야 한다.

클래스를 만들 때 고려해야할 점

1. 어떤 맴버 변수들이 있는가? -> 클래스가 어떤 상태(states)를 가질 수 있는가?

2. 특정 객체의 생성을 어떻게 구성하고 싶은가 -> 어떤 생성자를 사용할 것인가?

3. 어떤 행동을 실행할 것인가? -> 메서드를 어떻게 설계할 것인가?

생성자를 만들어 줄 때 속성 중에 바로 필요한 것이 있는지 없는지 찾아보고

확실하게 필요한 것들은 생성시에 바로 초기화한다.

-> 객체를 만들 때 확정해야 할 상태인지 그럴 필요 없는 지를 나눈다.

객체의 상태를 확인하고 싶다면 toString을 오버라이딩해서 확인하도록 하자.

String.format()을 사용하여 만들어줄 수 있다.

그 다음 객체에 허용하고 싶은 행위가 무엇인지를 생각해본다.

getter와 setter로 생각하는 방식보다는 사용자가 객체에 대해 어떻게 생각할지,

어떤 메서드가 객체를 사용하기 쉽게 할지 생각해야 한다.

사용자의 관점에서 어떻게 사용하고 싶을지를 생각해서 API를 작성한다.

객체를 설계할 때 중요한 것은 다른 객체와의 관계를 살펴보는 것이다.

예를 들어 Customer 클래스를 제작한다고 하자.

Customer가 가질 수 있는 상태로는 name, address 등이 필요할 것이다.

그리고 만일 위의 두 속성이 반드시 필요한 필수요소일 경우 생성자를 만들 때

초기화요소에 해당하지만 그렇지 않을 때는 초기화할 필요가 없다.

상속

상속은 Is A 관계가 있을 때 사용하는 것을 추천한다.

예를 들어 Person이란 클래스가 있고 Student라는 클래스가 있다고 하자.

Student is a person이다. 즉, person은 student를 포함하는 더 포괄적이고 상위 개념이다.

이 경우 student가 가져야할 속성을 person이 가지고 있기 때문에 상속의 효과가 크다.

하지만 이러한 포함 관계가 없다면(즉, Is A 관계가 아니라면) 필요없는 상태(속성)나 행동(메서드) 또한 상속하게 되기 때문에 비효율적인 상속이 된다.

자바에서는 다중상속을 지원하지 않는다.

물론 두 개의 인터페이스를 연장하는 것으로 다중상속이 가능하다고는 하나

C++의 다중상속과는 완전히 다르다.

다중상속은 프로그래밍의 어려움을 가져온다. 그렇기 때문에 자바는 다중상속을 지원하지 않는다.

하지만 연관관계의 연속성은 가질 수 있다. 즉, 상속 계층의 연장은 가능하다

예를 들어 Animal 클래스, Pet 클래스, Dog 클래스가 있을 때

public class Dog extends Animal, Pet {

...

}

과 같은 형태의 다중 상속은 불가능하지만

public class Pet extends Animal {

...

}

public class Dog extends Pet {

...

}

위와 같은 방식의 상속 계층(Animal <- Pet <- Dog)은 가능하다.