생성자 : 새로운 객체가 생성될 때 가장 먼저 일어나는 일 중 하나로 보통 프로그램의 초기화를 담당한다.

★ 객체 지향과 관계없이 항상 변수를 초기화 해주는 것이 좋다.

메서드 오버로딩 : 시그니처를 다르게 하여 다른 버전의 메서드를 제공가능

슈퍼클래스가 생성되는 방법

1. 생성자 안에서 클래스의 슈퍼클래스의 생성자가 호출된다.
2. 객체의 각 클래스 속성이 초기화된다.
   (클래스 정의의 일부인 속성)
3. 생성자의 나머지 코드가 실행된다.

오류처리 : 코드에 오류 조건을 감지하고 처리하는 기능

- 문제를 무시하는 건 좋은 생각이 아니다.
- 잠재적인 문제를 확인하고 문제를 발견하면 프로그램이 중단되도록 하자.
- 잠재적인 문제를 확인하고 실수를 파악한 후 문제를 해결하다
- 예외를 던진다. (try/catch)

클래스는 한개로 부터 여러 객체를 인스턴스화 할수 있다.
각 객체는 고유한 메모리를 가지게 되는데 일부 속성 및 메서드는
공유해서 사용한다.

메서드는 객체의 행위를 나타낸다.
객체의 상태는 속성으로 표현한다.

- 지역적인 속성 : 메서드 내의 지역 변수
- 객체의 속성 : 객체가 소유하고 있는 메서드 범위 밖
  클래스 범위 내의 변수
- 클래스의 속성 : static 등으로 선언된 전역 변수

연산자 오버로딩시 클래스를 사용하는 사람들이 혼동하지
않도록 문서화 하고 주석을 달아 의의를 환기시킨다.

다중 상속은 강력한 기능을 제공하지만 프로그래머와
컴파일러 작성자에게는 시스템의 복잡성이 크게 늘어나는 문제가 될 수 있다.

전체 복사: 모든 참조를 따르고 참조된 모든 객체에 대해
새 사본을 작성

단순 복사: 단순히 참조를 복사

결론

이번 장에서는 객체지향 개념에 대한 일반적인 이해에는 필수적이지 않지만, 클래스 설계와 같은 상위 수준의 객체지향 과제에 상당히 필요한 고급 객체지향 개념을 많이 다루었다. 4장 '클래스 해부하기'에서는 클래스를 설계하고 구축하는 방법을 구체적으로 살펴본다.