인터페이스 패턴 깃 정리 잘할 것

객체지향 프로그래밍이란

절차지향 프로그래밍의 경우 동일한 로직을 반복하려면 다시 코딩하거나 복사해야하는데

이 경우 코드의 길이가 길어져 비효율적이고 기존 코드를 재사용하는게 효율적 이때

함수(매소드)를 만들어 해당 로직을 매소드로 활용하는 것이 효율적

함수는 (매개변수를 받는데 이때 받는 매개변수로 함수를 구분함) = 오버로딩

클래스 매소드 모음

Static 매서드

메소드란? 작업을 수행하기 위한 명령문의 집합

어떤 값을 입력받아서 처리하고 그 결과를 돌려준다(입력 받는 값이 없을 수도 있고 결과를 돌려주지 않을 수도 있다.)

메소드의 장점과 작성지침 ; 반복적으로 수행되는 여러 문장을 메소드로 작성한다.

**접근제한자 [static] 리턴type 메소드명([매개변수1, 매개변수2,..]){**

**//return이 없을 경우 리턴type은 void**

**처리할 프로세스들**

**[return 리턴값;]**

메모리 영역상 힙메모리 영역에 클래스(설계도) 파일 상주

Stack 이라는 메모리영역에 실행대기

Static 함수는 기울임 표시, static키워드를 사용한 변수는 클래스가 메모리에 자동으로 올라서

인스턴트(객체) 없이 바로사용 가능

Static은 변수나 메소드에 키워드로 사용

kate의 좋아하는 커피만 "라떼"로 하였는데, kevin이 좋아하는 커피까지 "라떼"로 변경되어 버린 것이다.

kevin이 좋아하는 커피까지 변경되버린 이유는 coffee변수를 static으로 정의하였기 때문에 kate와 kevin 둘은 같은 coffee변수를 공유하는 것이다.

위와 같이 static은 결코 남발해서는 안되며 공통으로 값을 유지하고 싶을 때만! 사용해야 한다

Setter &getter

외부에서의 무분비한 데이터 설정을 막고 오류를 검증하기 위함?

Setter를 통해 클래스 내부의 private변수를 검증후 설정

Getter를 통해 privat

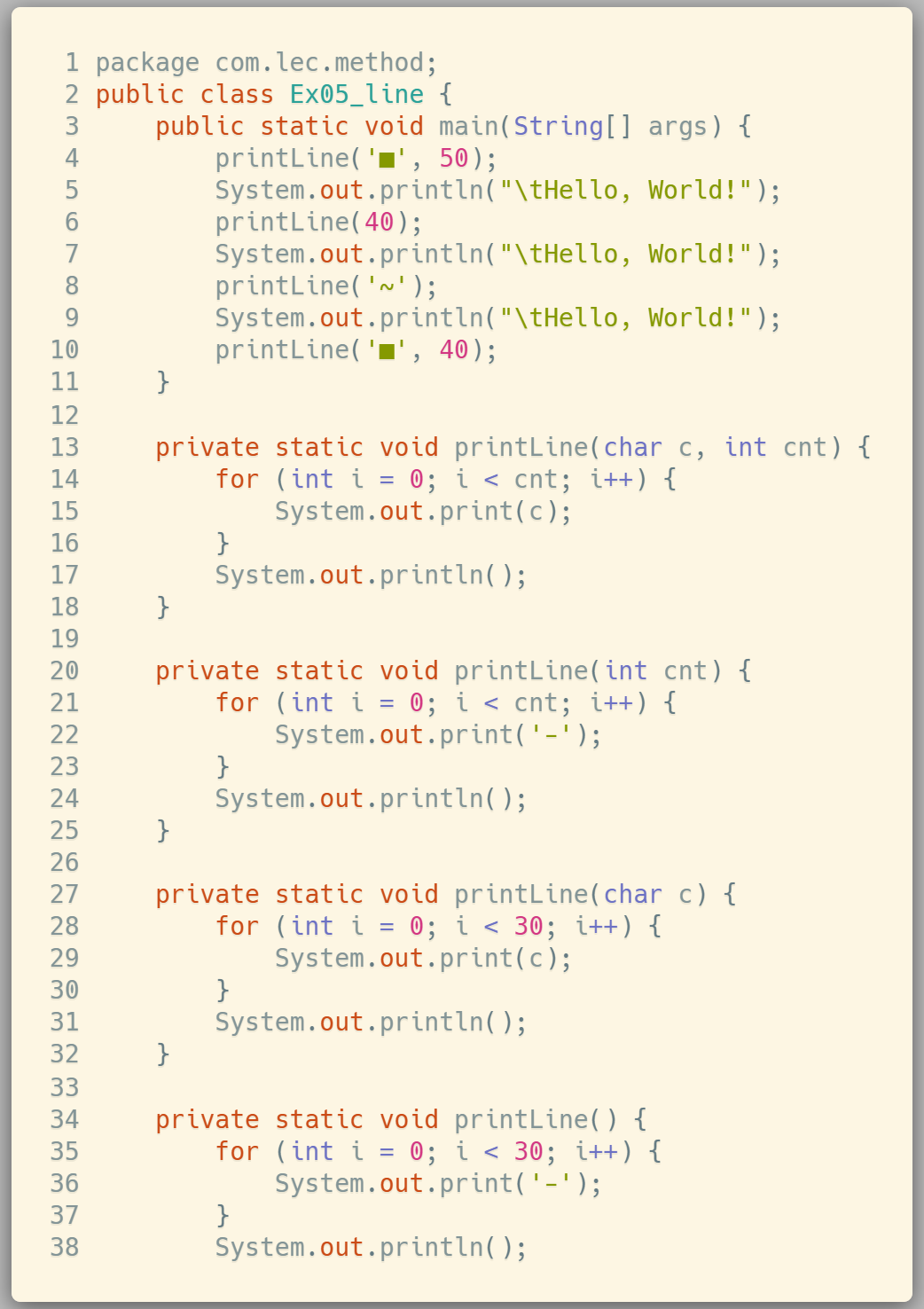
재귀적 함수호출

매소드 안에서 해당 매소드를 불러 반복적으로 작업하는 것?

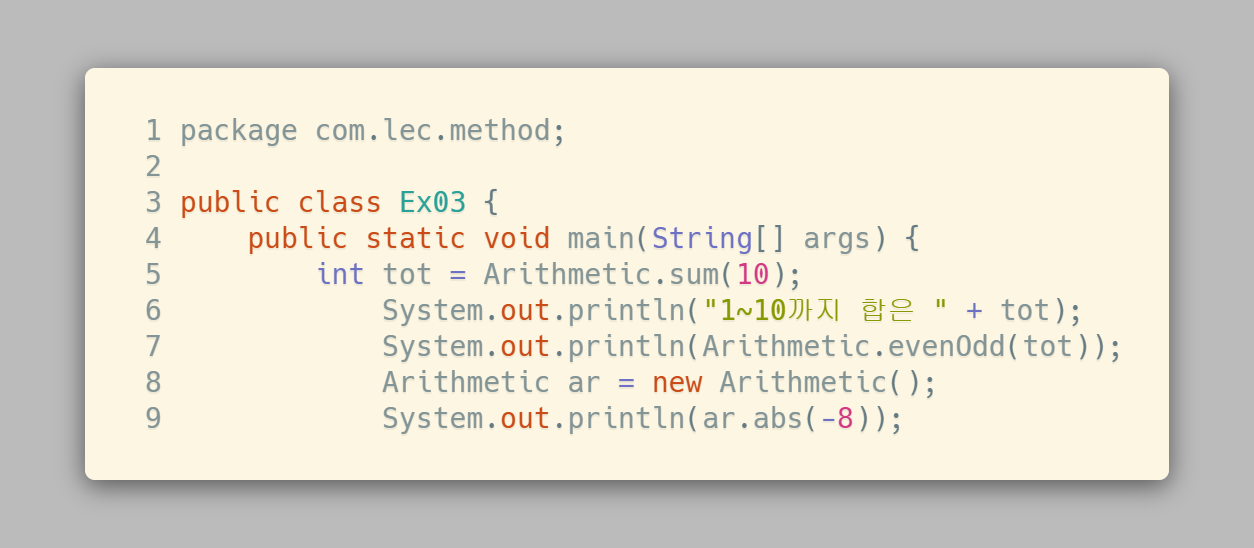
팩토리얼 알고리즘

리턴값

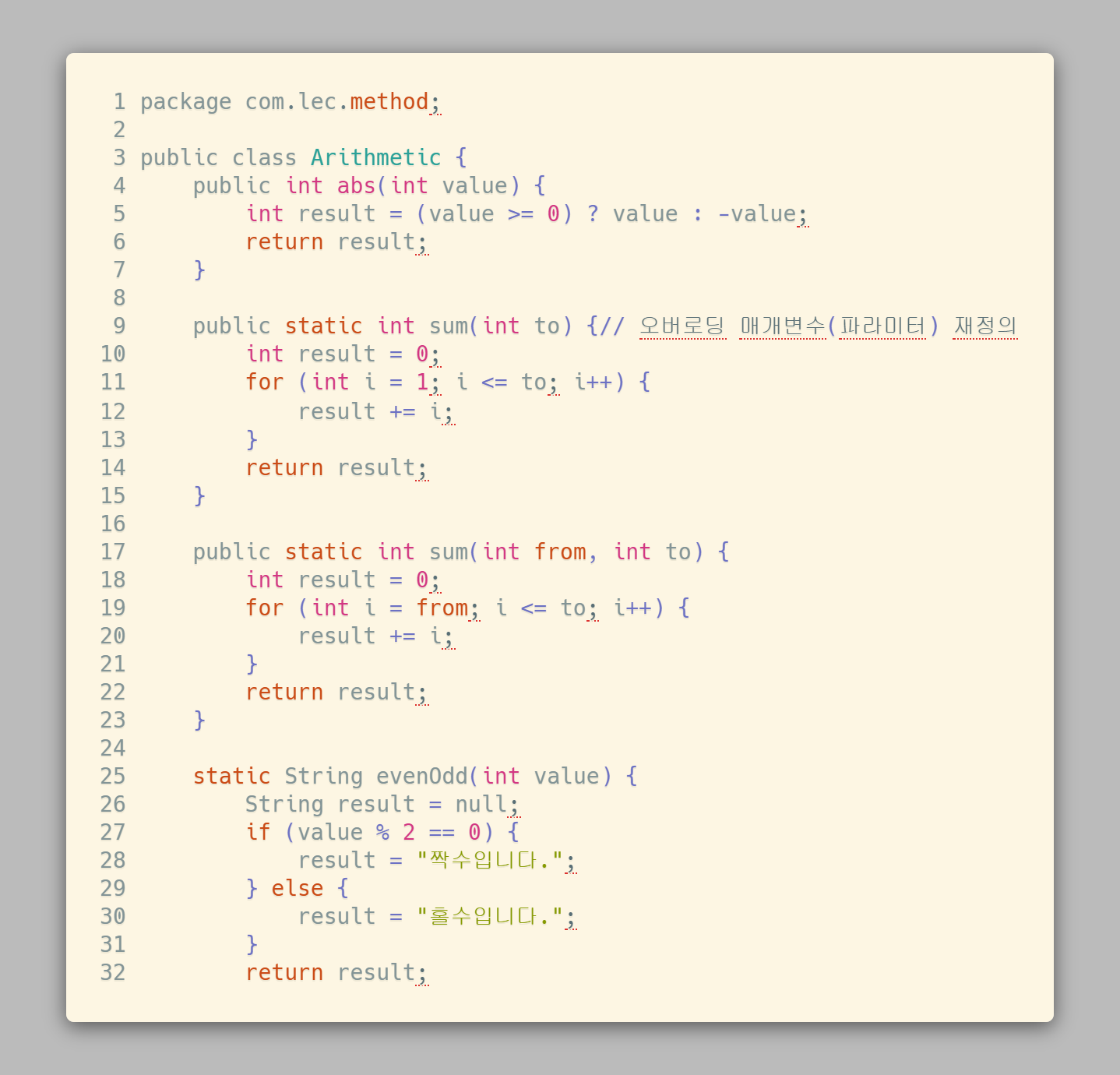
1. 메소드 프로그램 한계



여러 범위의 클라스 사용

****

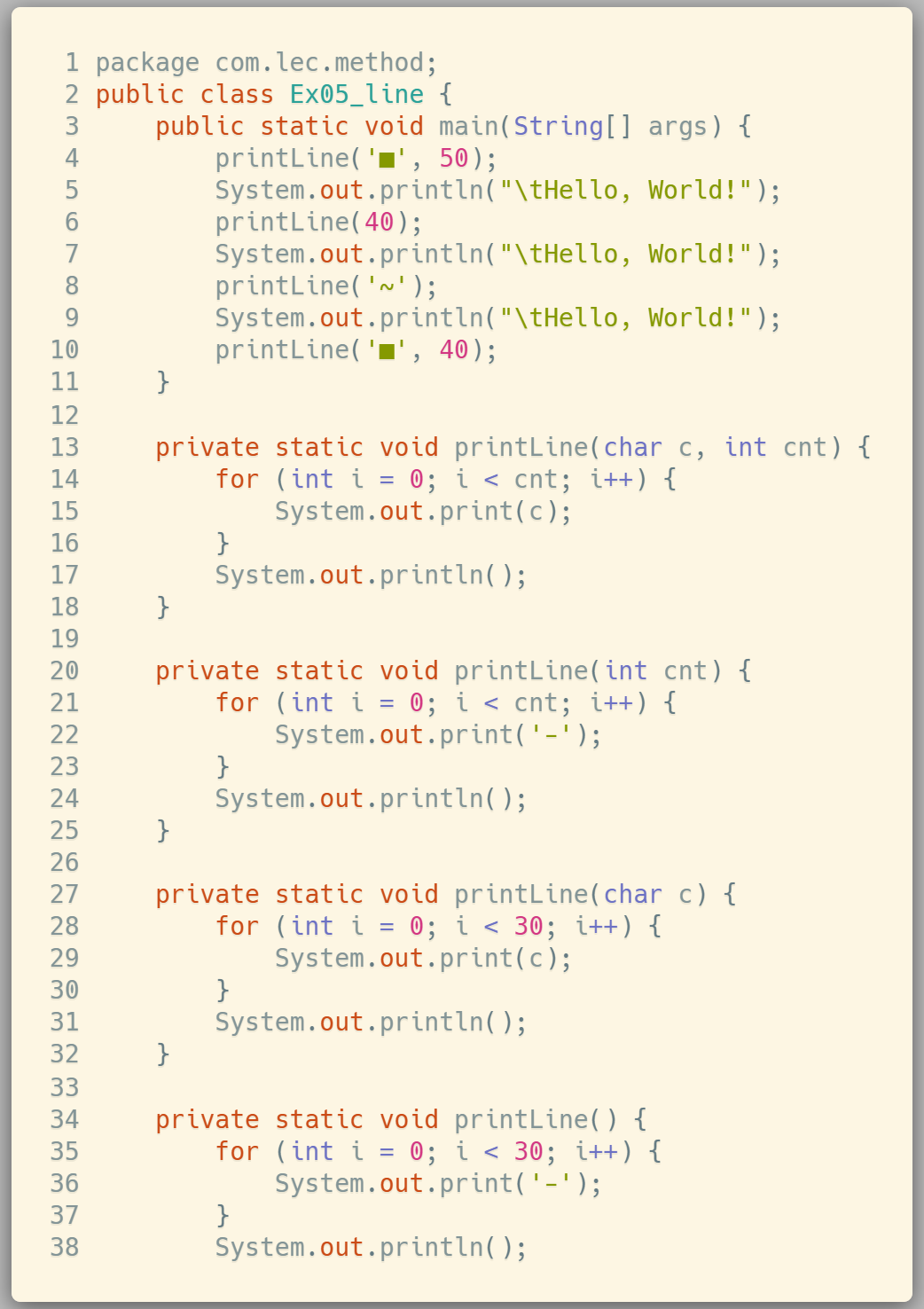
클래스 타입 및 사용 범위



팩토리얼 알고리즘



함수 오버로딩 매개변수 다르게



캡슐화 =

