객체지향설계및프로그래밍 실습과제#3

- 로봇 장난감을 클래스로 만들어 보기 –

# 0. 시작하기 전에

* 오늘부터 출제되는 실습과제들은 ‘우리 실습에서 새로 배우는 것들’ 위주로 구성되어 있으며, 따라서 **우리 모두에게 어렵습니다**. 최종 선택은 여러분의 판단에 달려 있겠지만, 조교는 여러분이 가급적 포기하지 말고 모든 부분과제를 도전해 보는 것을 권장합니다.

# Debugger 사용법 익히기 (60점)

조교와 함께 살펴본 Eclipse debugger 사용 방법을 토대로, Java project 하나를 골라 해당 프로그램이 어떻게 실행되는지 debugger를 통해 살펴보도록 합시다.

## 과제 수행 방법

* 디버그해 볼 Java project를 하나 고르세요.
  + 자신이 만든 Java project를 쓰는 것이 가장 좋지만, 적절한 것이 없는 경우 조교가 만든 OODP\_HW\_2\_3\_by\_TA.zip을 import해서 써도 좋습니다.
* 아래에 나열된 디버그 작업을 각각 한 번 이상 수행해 보세요:
  + main() 시작 부분에 breakpoint를 놓고 처음부터 한 줄 한 줄 단계별로 실행해 보기
  + main()이 아닌 다른 메서드의 시작 부분에 breakpoint를 놓고 실행한 다음,  
    해당 위치에서 실행이 중단되었을 때 Debug 탭 살펴 보기
* 실행이 어느 정도 진행되었을 때 각 변수 이름에 마우스 갖다 대 보기

## 과제 수행 조건

* 이 부분과제의 성공 여부는 여러분 스스로 판단하며, 조교는 여러분의 그런 선택을 존중합니다. 따라서, 이 부분과제는 조교에게 검사받지 않으며, 만약 자신이 이것만 하고 집에 가고 싶은 경우 느낀 점을 제출하고 바로 집에 가도 좋습니다.

# Toy class 만들어 보기 (30점)

이번 부분과제에서는 머리, 몸통, 다리가 N개 달린 장난감을 나타내는 Toy class를 만들어 보면서 Java에서 클래스를 다루기 위한 기본 문법들 / 개념들을 살펴 봅니다.

## 과제 수행 방법

* 중간 휴식시간 이후, 조교와 함께 전체 과제 내용을 함께 진행해 볼 예정입니다.
  + 이번 내용은 따라오기 쉽지 않으므로 별도 문서에 세세한 진행 순서를 기록해 두었습니다.
* 조교와 함께 전체 진행 순서를 모두 따라 오면, 이번 부분과제는 자동으로 완료됩니다!
* 완료되었다 하더라도, 아마도 여러분들 중 대다수는 우리가 만든 Toy class를 마음에 들어 하지 않을 것입니다. 자신이 여기에 뭔가 더 멋진 아이디어를 담고 싶다면, 조교의 도움을 받아 가며 좀 더 실제 장난감과 유사한 느낌이 나도록 Toy class를 개선해 보도록 합시다.

## 과제 수행 조건

* 전체 진행 순서를 완료하고 코드 작성을 끝내면 조교에게 검사를 요청하세요. 조교는 여러분의 Toy class가 기본적인 구현 조건(진행 순서에서 언급되었던)들을 만족하는지 살펴 보고 여러분이 새로 부여한 아이디어들을 살펴 볼 것입니다.

## 주의사항

* 이제까지 몇 번 느껴왔겠지만, Java 문법은 그리 호락호락하지 않습니다. 생각은 잘 나는데 이걸 Java로 어떻게 표현할 지 모르겠으면 지체하지 말고 조교의 도움을 요청해 주세요.

# ‘장난감 세계’와 ‘장난감 공장’ 개념을 더 도입해 보기 (10점)

우리 실습의 두 번째 도전과제는, 이전까지 만들었던 ‘static이 붙은 것’들과 오늘 배운 ‘static 없는 것’들을 함께 사용하는 방법을 살펴 보고, ‘public이 붙은 것’들과 ‘public 없는 것’, 그리고 ‘private이 붙은 것’의 의미를 되새겨 보는 것이 주요 목표입니다. 이전 과제들에 비해 생각할 내용이 좀 많지만, 구현 난이도 자체는 그리 어렵지 않으니 너무 걱정하지는 마세요.

## 필요한 것

* 새 ‘package’ 만드는 방법
  + 새 class 만들 때와 유사하게, 우클릭🡪New🡪package 누르고 이름 적어서 엔터 치면 됩니다.
* 어떤 class를 어떤 package 안에 정의하는 방법
  + 새 class 만들 때 특정 package를 우클릭하여 만들면 그 package 안에 정의됩니다.
  + 또는, Eclipse의 Package Explorer(보통 코드 창 왼쪽에 있는)에서 원하는 class 코드 파일을 원하는 package로 드래그해서 넣어 주면 됩니다. 이 때 번거로운 오류가 발생할 수 있으니, 가급적이면 class 만들 때 미리 이 class가 들어갈 package에 만드는 편이 좋습니다.
* private 키워드 사용 방법
  + 이번에 만들 필드들 중 일부는 ‘다른 어떤 class들도 못 보게’ 만들 필요가 있습니다. private 키워드는 이럴 때 사용되며 public이 붙는 자리(선언의 가장 왼쪽)에 붙여 주면 됩니다.
* 위의 내용을 포함하여 이번 과제에서 필요한 것들 중에서 package 관련 부분을 제외하면 모두 우리 수업(이론, 실습 모두 포함)에서 한 번 이상 다루어 본 적이 있습니다. 만약 자신이 이번 과제를 스스로 완성해 보고 싶다면 이전 수업자료들을 한 번 열어 보는 것도 좋겠습니다.

## 과제 진행 순서

* 새 프로젝트를 만들고, 평소처럼 Program class와 main()을 만듭니다.
* ‘장난감 세계’에 해당하는 ‘toys’ package를 만들어 줍니다.
* toys package 안에 Toy class와 ToyFactory class를 만듭니다.
  + Toy class는 이전 부분과제의 것과 동일하게 ‘장난감 하나’를 나타냅니다.
  + ToyFactory class는 새 장난감을 ‘일정한 불량률 아래 랜덤으로’ 만들어 주는 static 메서드를 제공하는 클래스입니다.
* Toy class에 필드, 생성자, toString()을 추가해 줍니다.
  + 이 때 필드는 public 없이 선언하여 장난감 세계 밖에서 마음대로 바꿀 수 없도록 만듭니다.
* ToyFactory class 안에 private Random rand = new Random(); 을 추가해 줍니다.
  + rand의 경우 장난감 공장의 생산과 밀접하게 연관된 중요한 필드이므로 private으로 둡니다.
* ToyFactory class에 public static Toy MakeOne() 을 구현해 줍니다.
  + 이 메서드는 새로운 Toy instance를 만들어 반환해 주는 메서드입니다.
  + 이 때, 30% 확률로 불량품을 반환하도록 if문과 창의력을 적절히 섞어 줍시다.  
    (어떤 것이 ‘불량품’인지는 여러분이 자유롭게 지정해 주세요)
* Toy class에 ‘내가 불량품인지’ 반환하는 boolean IsValid() 를 구현해 줍니다.
* Toy class에 ‘불량 상태를 복원’하는 void Fix() 를 구현해 줍니다.
* 이제 main()에서 아래의 작업을 순서대로 수행해 봅니다.
  + 10칸짜리 Toy 배열을 만듭니다.
  + for문을 통해 배열의 각 칸을 ToyFactory.MakeOne()을 호출하여 채워 줍니다.
  + foreach문과 sysout을 통해 배열 안의 장난감들을 출력해 봅니다.  
    (출력 내용을 읽기 편하게 foreach문 위/아래에서 적절한 메시지와 줄바꿈을 출력합시다)
  + foreach문과 IsValid()를 통해 배열 안의 불량 장난감들에 대해 Fix()를 호출해 줍니다.
  + 다시 배열 안의 장난감들을 출력하여 복원이 잘 되었는지 확인해 봅니다.
  + 이제 오늘 할 일은 끝났으니 적절한 완료 메시지를 출력하고 실행을 종료합니다.

## 과제 수행 조건

* 기본적으로 이번 부분과제는 위의 진행 순서를 모두 준수하여 프로그램을 작성해야 합니다.
* 지난 실습과제#2-3과 동일하게, 과제의 성공 여부 판단은 조교가 수행합니다.  
  이번 부분과제의 경우 다양한 곳에서 다양한 키워드들을 쓰고 있으며,  
  조교는 검사 과정에서 해당 키워드의 의미와 ‘여기 붙은 이유’를 물어 볼 예정입니다.  
  물론 틀려도 집에 못 가는 것은 아니니 걱정 말고 조교와 즐거운 토론을 진행해 봅시다.

**주의사항**

* 지난 시간 내용을 아직 덜 이해한 경우 이 부분과제는 생각보다 어려울 수 있습니다. 그렇다고 무작정 포기하지만은 말고 자신이 이 부분과제에 도전해 보기를 희망하는 경우 조교의 도움을 받아 가며 차근차근 진행해 보는 것을 권장합니다.