Injector

- Необходимо реализовать статический метод `Injector.initialize`, эмулирующий работу контейнера Dependency Injection.
- Его основная задача - создать объект класса, полное имя которого передано в параметре `rootClassName`.
- Также в метод передается набор доступных классов реализаций. Для каждого такого класса за один `initialize` должно быть создано не более одного экземпляра.
- Гарантируется, что каждый класс содержит ровно один конструктор.

Injector

Параметры конструктора определяют зависимости данного класса.

```
public class A {
    public A(Interface1 x) {}
}

public class B implements Interface1 {}
```

Класс A зависит от интерфейса `Interface1` (так же зависимостью может быть абстрактный или конкретный класс).

Таким образом, чтобы создать объект класса `A`, можно сначала создать объект класса `B` и передать его в конструктор.

Понятно, что зависимости могут быть несколько более замысловатыми.

Injector

- Метод `initialize` должен завершиться с исключением `AmbiguousImplementationException`, если найдено несколько разных классов-реализаций одной и той же зависимости (зависимости --- это только те классы и интерфейсы, которые встречаются в конструкторах).
- Метод `initialize` должен завершиться с исключением `ImplementationNotFoundException`, если для какой-то зависимости не найдено ни одной реализации.
- Метод `initialize` должен завершиться с исключением `InjectionCycleException`, если зависимости прямо или косвенно образуют цикл.
- В остальных случаях метод должен вернуть объект класса `rootClassName`.

Формат сдачи

- Запаковываете проект в zip-архив
- Посылаете письмо преподавателю с архивом:
  - s-proshev@ya.ru
  - Java02. КР 02 <фамилия> <имя>