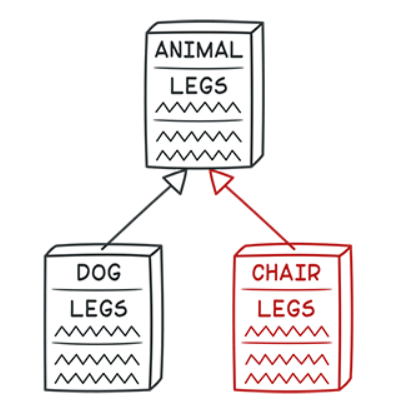
Answers-Kulina Preliminary Test-Data Scientist

By: Muhammad Murtadha Ramadhan

1. **Basic Concepts**
   1. **Contoh code smell pada OOP: ‘Refused Bequest’**

Hal tersebut terjadi ketika *subclass* yang dibentuk hanya menggunakan beberapa properti dan method yang diturunkan dari *superclass*-nya. Hal tersebut mengindikasikan bahwa terdapat properti dan method yang tidak terpakai. Biasanya hal ini terjadi karena seorang *programmer* ingin membuat *inheritance* dengan motivasi untuk melakukan *code reuse* dari *superclass* yang digunakan. Contoh dapat diilustrasikan pada gambar berikut:



Source: https://refactoring.guru

Ilustrasi tersebut menggambarkan bahwa *subclass* CHAIR dapat menggunakan properti LEGS dari *superclass* ANIMAL. Akan tetapi, *subclass* CHAIR dan *superclass* ANIMAL merupakan objek yang berbeda dan tidak mempunyai keterhubungan satu sama lain.

**Treatment:**

Apabila *inheritance* tidak memungkinkan serta *subclass* tidak mempunyai hubungan dengan *superclass*, *inheritance* dapat dihapus dan digantikan dengan metode pendelegasian yang mana objek *superclass* ditaruh di dalam suatu *field* sehingga dapat metode tertentu dalam *superclass* dapat digunakan.

* 1. **Dependency Injection Concept**

Konsep Dependency Injection digunakan dengan membuat suatu *class* semisal *Injector* untuk memberikan service yang dibutuhkan oleh objek *Client*. Dari hal tersebut, *Client* tidak perlu menginstantiasi object *Service* pada class-nya dan independen dari pembuatan objek *Service* yang dibutuhkan. Keuntungan dalam penggunaan Dependency Injection yaitu:

* + - 1. Depedency injection memberikan fleksibilitas pada *Client* untuk dikonfigurasi.
      2. Dependency injection mengurasi posibilitas *coupling* antara sebuah kelas dan dependensinya.

1. **Do and Don’t REST API**
   * 1. GET

**Do:** Gunakan metode GET untuk mengambil data atau informasi saja

**Don’t:** Jangan menggunakan GET untuk mengubah suatu *state*

* + 1. POST

**Do:** Gunakan HTTP *response code* 200 untuk ‘OK’ dan 204 untuk ‘no content’ untuk melihat *response* dalam penggunaan POST

**Don’t:** Jangan melakukan dua request POST yang identik karena dapat menyebabkan terdapat 2 resource yang berbeda yang menyimpan informasi yang sama atau redundansi.