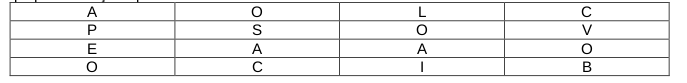
# Lista de Exercícios para a N1 - Engenharia de Software III

### 39- A partir do caça objetos abaixo, encontre no mínimo cinco objetos e modele uma relação todo-parte, abstraindo o nome de uma ave na classe todo e os cinco objetos encontrados como partes deste todo. Explique a solução apresentada.



Saco

Boia

SAPO

BOI

VACA

LEAO

CASACO

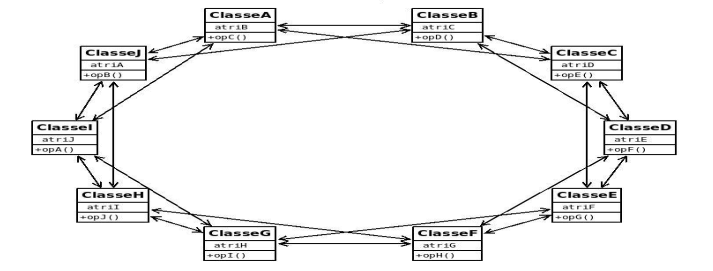
SOCO

BOLO

ESPACO

nao entendi esse exercicio

### 40- Analise o diagrama de classes abaixo. Descreva a coesão e acoplamento desse diagrama. Modele um diagrama de classes que seja melhor em termos de coesão e acoplamento e explique o porquê do novo diagrama de classes ser mais vantajoso do que o abaixo.



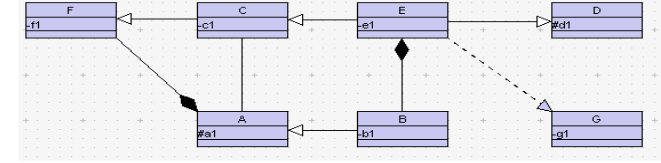
Todas as classes desse diagrama apresentam alto acoplamento e baixa coesão. Isso significa dizer que as classes possuem múltiplas dependências umas das outras, então se uma delas for modificada, todo o ciclo vai ser quebrado, pois uma classe sofreu alteração. Para um projeto profissional, de negócios, é uma situação inapropriada. Imagine que está trabalhando em um projeto com mais de 100 classes interdependentes (alto acoplamento) e precisa executar alteração em uma delas. Todo o projeto necessitará de refatoração e isso, numa empresa, envolve tempo e desperdício de tempo é desperdício de dinheiro. Devemos, então, aumentar a coesão e diminuir o acoplamento para que as classes possam ser modificadas sem maiores influências em um grande número de outras classes.

O diagrama remodelado seria, mais ou menos, dessa forma:

Precisamos inserir moderadores de acesso (public, private, protected e default)

FALTA REMODELAR ESSE DIAGRAMA

### 41 - Identifique cada um dos relacionamentos utilizados no diagrama de classes abaixo, inclusive se são simétricos ou assimétricos. Quais atributos a instância de cada classe pode visualizar? Justifique a tua resposta.



Relacionamentos:

Dependência: Uma classe depende da outra para que toda a sua funcionalidade seja implementada corretamente.

E - G → dependência. E necessita de G.

Composição: Quando o container é destruído, o conteúdo também é.

Então, no caso, as relações de composição presentes entre as classes:

A - F → A possui F. Se A é destruída, F também é

E - B → E possui B. Se B é destruída, E também é.

Generalização/Especialização: representa a herança entre classes.

B - A → Herança - B herda de A.

F - C → Herança - C herda de F.

E - C → Herança - E herda de C.

E - D → Herança. E herda de D.

A classe E possui herança múltipla.

Visibilidade de atributos:

B - A → como B herda de A, B tem permissão para visualizar os atributos protegidos de A.

E - D → como E herda de D, E tem permissão para visualizar os atributos protegidos de D.

Todos os outros atributos de outras classes são visíveis apenas para a própria classe, visto que são atributos privados.