Objetivo 3: Agregação, Composição e outras associações
Lista 1

Objetivos

1. Assimilar os conceitos de outras associações entre classes, como, agregação, composição, associação binária, e classe associativa;
2. Entender a relação entre objeto-todo e objeto-parte;
3. Entender como programar as relações de agregação e composição;
4. Entender como programar as relações binárias;
5. Entender como programar as classes associativas;
6. Interpretar diagramas de classe da UML para transformá-los em código na tecnologia Java;
7. Criar pequenos apps Java em modo texto.

EXERCÍCIO

1. Crie um projeto na IDE Eclipse (dica de nome: Objetivo3_Lista1) na arquitetura de software MV e faça o que se pede a seguir:

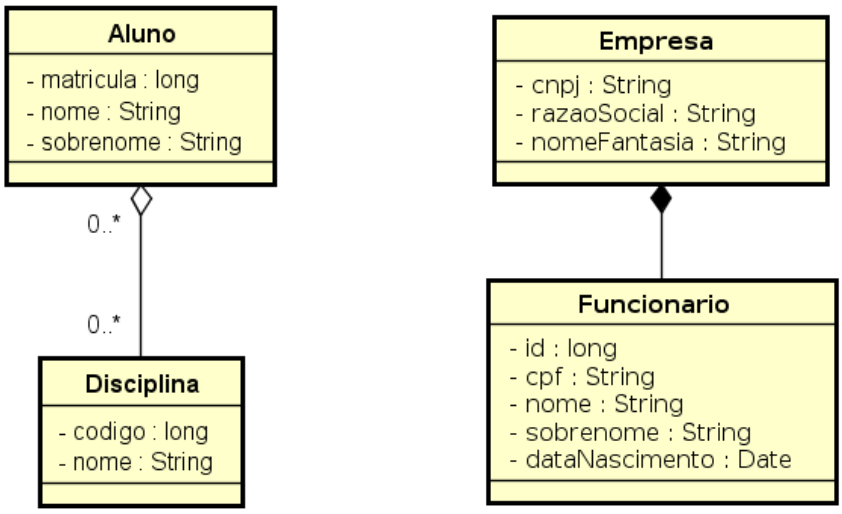
a. No pacote model você deverá colocar a interpretação das classes dos diagramas das Figuras abaixo. O diagrama Usuario-Corrida-Motorista-Veiculo foi criado por um colega nosso e não foi revisado, logo, pode conter erros que você deve corrigir ao interpretar o diagrama. Um exemplo está no atributo anoFabricacao, da classe Veiculo, ele está como *Date*, onde, deveria vir como *Integer*. As associações também devem ser objeto de revisão, principalmente no que tange a sua multiplicidade. Se encontrar algum erro, o corrija na implementação.

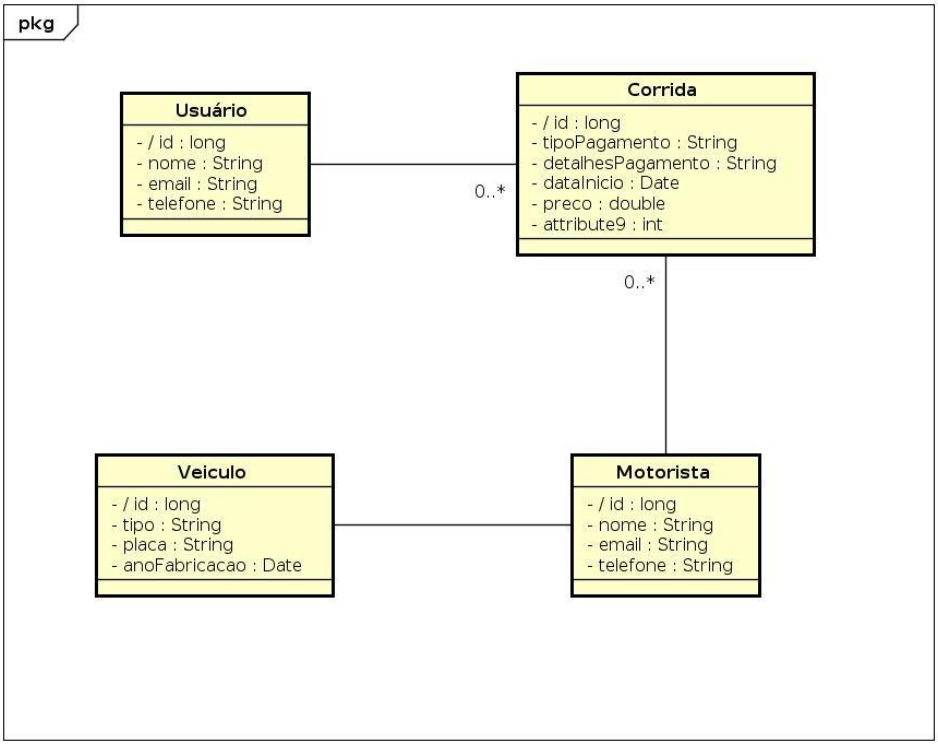
b. Crie um *controller* para o diagrama Aluno-Disciplina, nele crie dois objetos da classe Aluno, a1 e a2, e três objetos da classe Disciplina, d1, d2 e d3, todos com valores válidos nos seus atributos. Matricule o a1 em duas disciplinas e o a2 em todas as disciplinas. Depois, faça o programa imprimir todas as disciplinas por aluno, ordenado pelo critério nome da

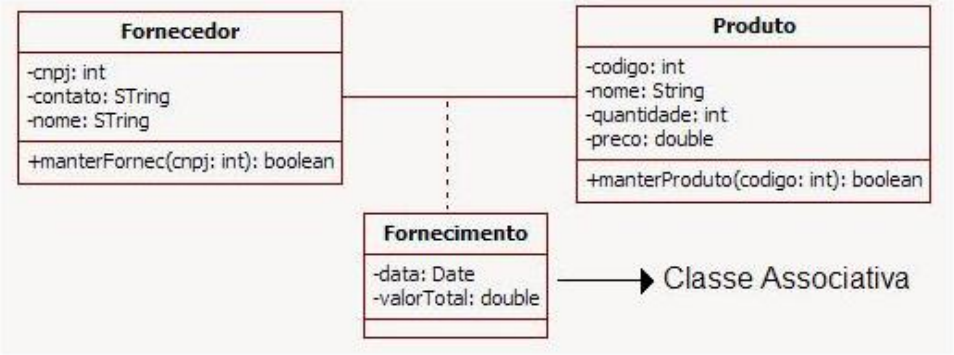
disciplina, em ordem crescente. Agora desfaça a matricula do a2 das disciplinas e reimprima as disciplinas por aluno, mesmo critério anterior.

c. Crie um *controller* para o diagrama Empresa-Funcionario, nele crie seis objetos da classe Funcionário, f1, f2 ... f6, e dois objetos da classe Empresa, e1 e e2, todos com valores válidos nos atributos. Inscreva os funcionários f1, f2 e f3 na e1, e os f4, f5 e f6 na e2. Depois, faça o programa imprimir todos os funcionários por empresa, critério nome, ordem crescente. Agora faça o programa imprimir os funcionários de cada empresa, ordenados pelo critério da idade, na ordem decrescente.
d. Baseado no diagrama Usuario-Corrida-Motorista-Veiculo, um Uber básico, crie um *controller*, e crie os objetos necessários com valores válidos em seus atributos, e faça o programa realizar duas corridas para um mesmo usuário, com um mesmo motorista e veículo. Faça o programa as imprimir todas as **corridas por usuário**, critério dataInicio, em ordem decrescente. Depois, faça o programa imprimir todas as **corridas por motorista**, critério dataInicio, ordem decrescente. Agora faça o programa calcular o **valor total** de todas as corridas **por motorista**.
e. Baseado no diagrama Fornecedor-Fornecimento-Produto, crie um *controller*, e crie os objetos necessários, com atributos válidos, para realizar o fornecimento de três produtos diferentes, de dois fornecedores diferentes, por exemplo, dois produtos de um mesmo fornecedor e um produto do outro fornecedor. Coloque todos os fornecimentos em uma coleção e faça o programa os imprimir os fornecimentos e o total do que foi fornecido.

Bom trabalho.


Aluno
- matricula : long
- nome : String
- sobrenome : String
0..*
0..*
Disciplina
- codigo : long
- nome : String
Empresa
- cnpj : String
- razaoSocial : String
- nomeFantasia : String
Funcionario
- id : long
- cpf : String
- nome : String
- sobrenome : String
- dataNascimento : Date


pkg
Usuário
- / id : long
- nome : String
- email : String
- telefone : String
0..*
Corrida
- / id : long
- tipoPagamento : String
- detalhesPagamento : String
- dataInicio : Date
- preco : double
- attribute9 : int
0..*
Veiculo
- / id : long
- tipo : String
- placa : String
- anoFabricacao : Date
Motorista
- / id : long
- nome : String
- email : String
- telefone : String



Gabarito

Você encontra o gabarito deste exercício no repositório lpoo do github do professor.

Referências Bibliográficas

DEITEL, Paul; DEITEL Harvey. **Java: como programar**. 8. ed. Rio de Janeiro: Pearson, 2010.

GUEDES, Gilleanes T. A. **UML 2: uma abordagem prática**. 2 ed., São Paulo: Novatec Editora, 2011.