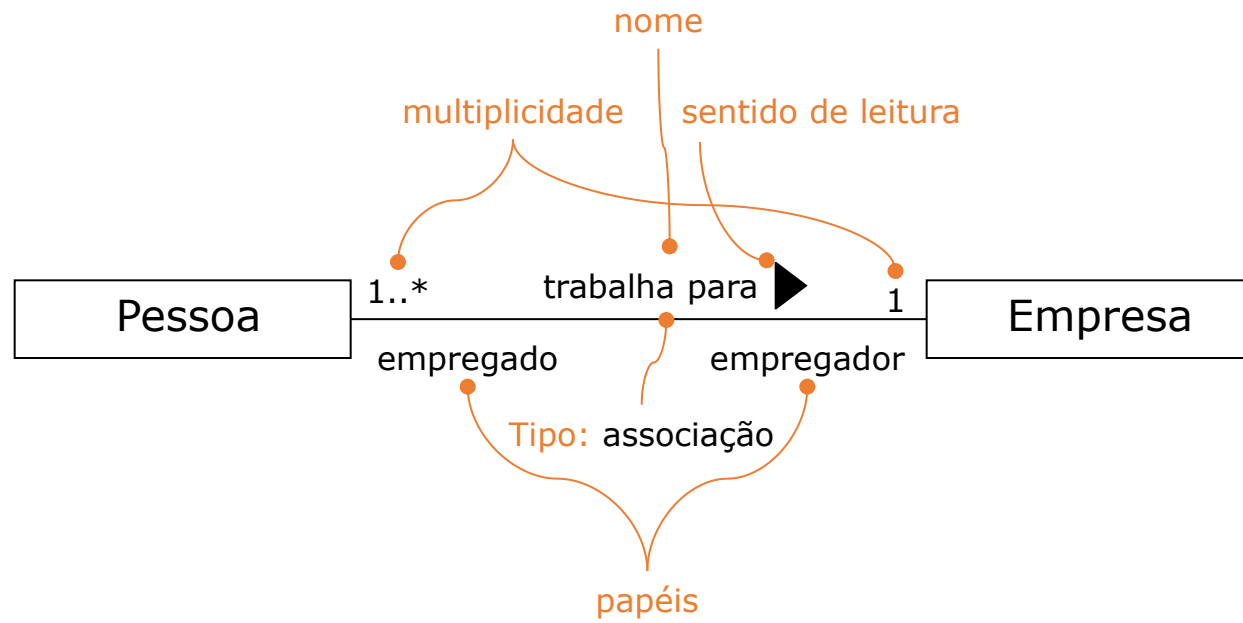


Diagrama de Classes - Relacionamentos

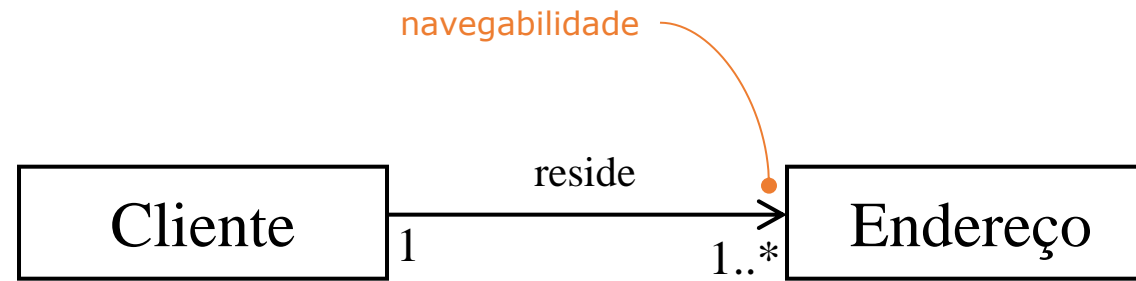
- Os relacionamentos possuem:
 - **Nome:** descrição dada ao relacionamento (faz, tem, possui,...)
 - **Sentido de leitura**
 - **Navegabilidade:** indicada por uma seta no fim do relacionamento
 - **Multiplicidade:** 0..1, 0..*, 1, 1..*, 2, 3..7
 - **Papéis:** desempenhados por classes em um relacionamento
 - **Tipo:** associação (agregação, composição), generalização e dependência
 - Vamos detalhar mais adiante

Relacionamentos



E a navegabilidade?

Relacionamentos



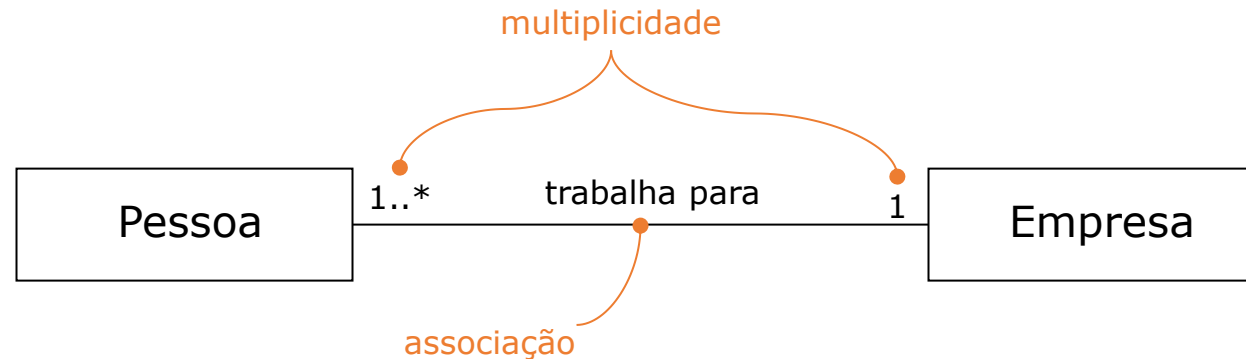
- O cliente sabe quais são seus endereços, mas o endereço não sabe a quais clientes pertence

Tipos de Relacionamento

- Associação
 - Agregação
 - Composição
- Generalização / Especialização
- Dependência

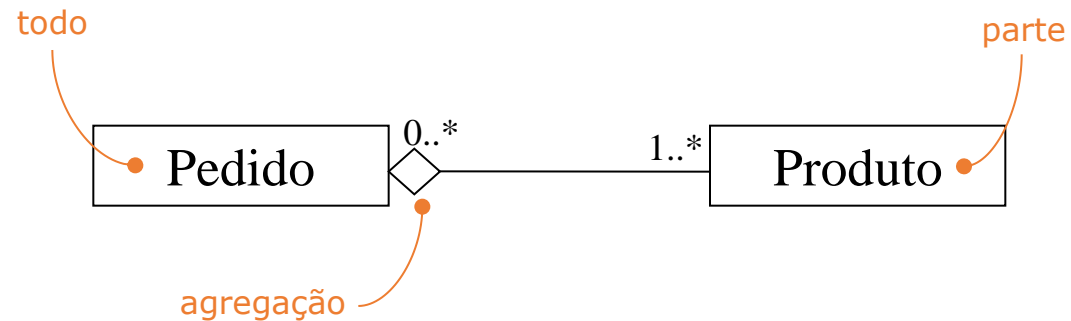
Associação

- Uma **associação** é um relacionamento estrutural que indica que os objetos de uma classe estão vinculados a objetos de outra classe.
- Uma associação é representada por uma linha sólida conectando duas classes.



Agregação

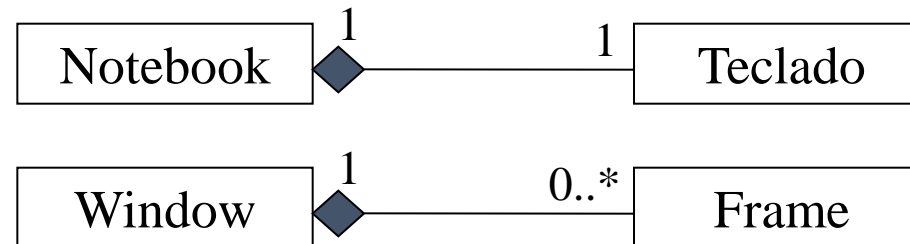
- É um tipo especial de associação
- Utilizada para indicar “todo-parte”



- um objeto “parte” **pode** fazer parte de vários objetos “todo”

Composição

- É uma variante semanticamente mais “forte” da agregação
- Os objetos “parte” só podem pertencer a um único objeto “todo” e têm o seu tempo de vida coincidente com o dele

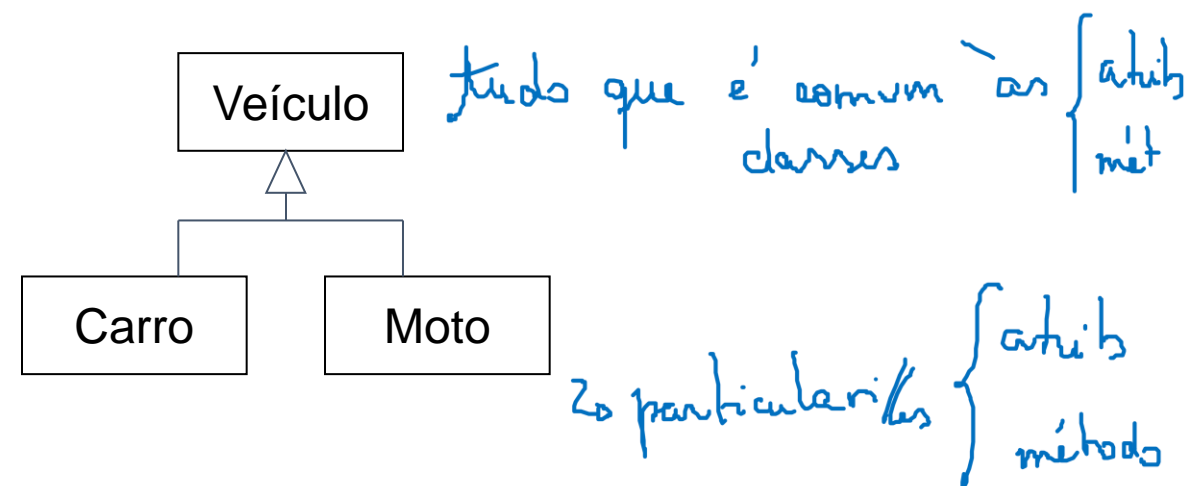
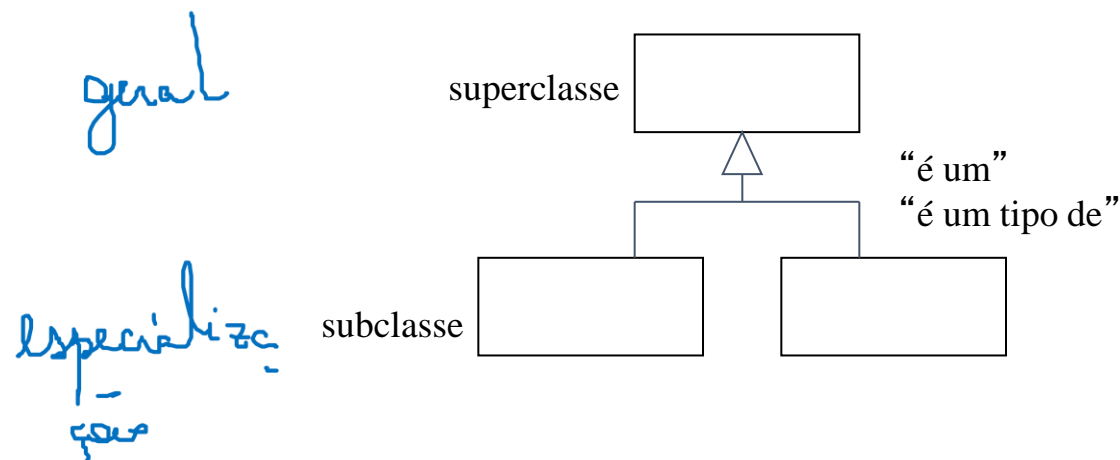


- Quando o “todo” *morre* todas as suas “partes” também *morrem*

Generalização / Especialização = Herança

- É um relacionamento entre itens gerais (superclasses) e itens mais específicos (subclasses)

Reutilização
de
Código



Dependência

- Representa que a alteração de um objeto (o objeto independente) pode afetar outro objeto (o objeto dependente)

Ex:



Obs:

- A classe cliente depende de algum serviço da classe fornecedor
- A mudança de estado do fornecedor afeta o objeto cliente
- A classe cliente não declara nos seus atributos um objeto do tipo fornecedor
- Fornecedor é recebido por parâmetro de método

Classe

atributos = características

metodos = comportamentos

alteram o estado

Pessoa

- nome

- idade

+ setNome(n)

+ fazAniversario()

Pessoa p1 = new Pessoa();

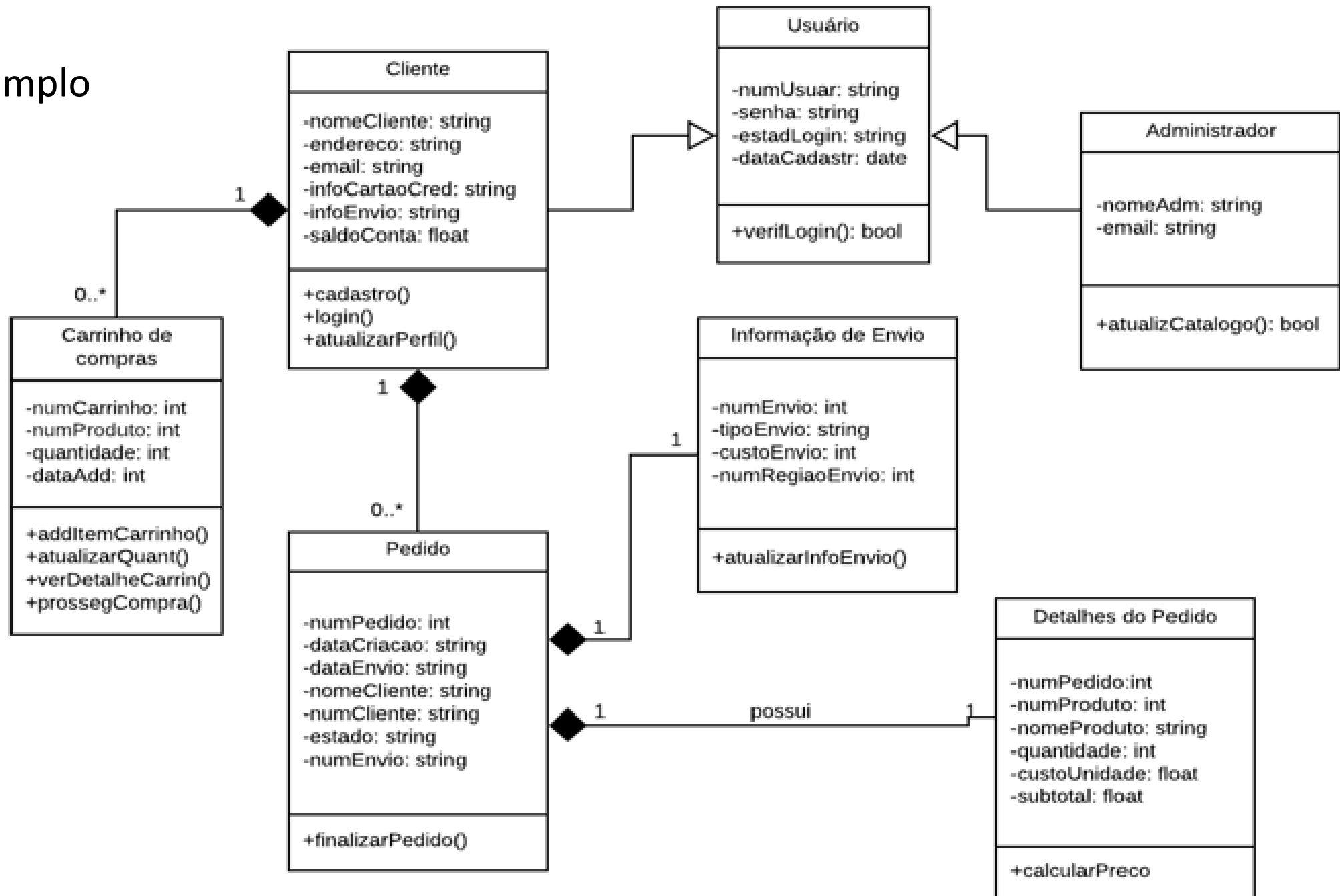
p1.setNome("Ana");

p1.fazAniversario();

p1 =>

nome = Ana
idade = 1

Um exemplo



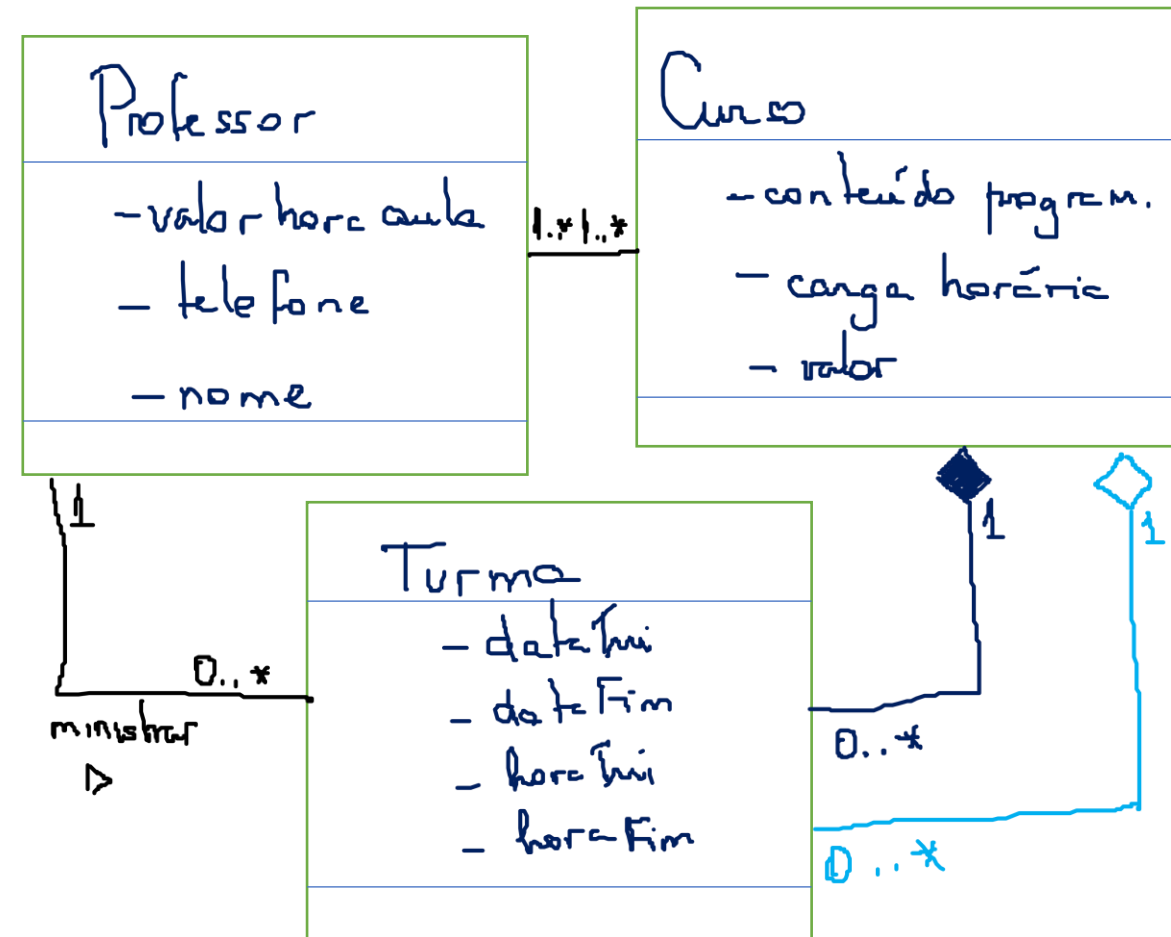


Bora praticar

Cenário 1

Uma empresa oferece cursos profissionalizantes

- O dono precisa controlar
 - os cursos oferecidos, as datas de início e de término, os horários de início e término
 - o nome e o telefone celular do professor e o valor da sua hora/aula
 - para cada curso, deve-se controlar
 - a carga horária, o conteúdo programático e o valor do curso



Cenário 2

Gerenciamento de locação de festas

- É necessário controlar os aluguéis, para isso é preciso cadastrar:
 - o nome e o telefone do cliente
 - o endereço completo da festa
 - o tema escolhido
 - a data da festa, os horários de início e fim da festa
- Em alguns casos, são oferecidos descontos. Sendo assim, é preciso saber o valor realmente cobrado num determinado aluguel, isto é, para cada tema, é preciso controlar:
 - a lista de itens que compõem o tema (ex: castelo, boneca da Cinderela, bruxa etc.)
 - o valor do aluguel
 - a cor da toalha da mesa que deve ser usada com o tema

É necessário controlar os aluguéis, para isso é preciso cadastrar:

- o nome e o telefone do cliente
- o endereço completo da festa
- o tema escolhido
- a data da festa, os horários de início e fim da festa

Em alguns casos, são oferecidos descontos. Sendo assim, é preciso saber o valor realmente cobrado num determinado aluguel, isto é, para cada tema, é preciso controlar:

- a lista de itens que compõem o tema (ex: castelo, boneca da Cinderela, bruxa etc.)
- o valor do aluguel
- a cor da toalha da mesa que deve ser usada com o tema

Rascunho

