Engenharia e Requisitos de UML – 03/10/2022

**Classes de objetos**

Classes: podemos dizer que as classes representam uma categoria (molde).

Objetos: elementos criados a partir da classe (objetos criados a partir do molde).

OBS: instanciar é produzir, criar.

**Atributos ou propriedades**

Representam as características de uma classe. São informações que costumam variar de um objeto para outro e que permitem diferenciar um objeto de outro da mesma classe.

Nome: tipo de dado armazenado.  
Tipos de dado:  
int – valores inteiros;  
float – valores reais;  
string – palavras, frases, caracteres;

|  |
| --- |
| Atributos da classe Bombom  Nome: tipo de dado armazenado sabor: string recheio: string Peso: float dataFabricacao: date. |

**Métodos, operações ou comportamentos**

Um método representa uma atividade que um objeto de uma classe pode executar.  
O método pode, ou não, receber **parâmetros** (pré-requisitos para que o método posso ser executado).  
O método pode, ou não, retornar valores (resultado da ação).

|  |
| --- |
| Nome() |

Um método representa uma atividade que um objeto de uma classe pode executar.

|  |
| --- |
| Métodos  derreterChocolate() aplicarNasFormas() aplicarRecheio() embalar() |

**Visibilidade**

A visibilidade indica o nível de acessibilidade de um determinado atributo ou método, sendo representada à esquerda destes.

|  |
| --- |
| **(-) privada** – significa que somente os objetos da classe detentora do atributo ou método poderão enxerga-lo ou utilizá-lo **(#) protegida** – determina que atributos e (ou) métodos poderão ser acessados pela classe detentora e também por suas subclasses. **(+) pública** – atributo ou método pode ser acessado por qualquer objeto. **(~) pacote** – determina que o atributo e(ou) método é visível para qualquer objeto dentro do pacote. |

No exemplo abaixo, os atributos foram definidos como tipo “privado” e os métodos como público.

|  |
| --- |
| Bombom  - sabor: string - recheio: string - peso: float - dataFabricacao: date  + derreterChocolate() + aplicarNasFormas() + aplicarRecheio() + embalar() |

**Herança**

Permite o reaproveitamento de atributos e de métodos otimizando o tempo de desenvolvimento, além de permitir a diminuição de linhas de código e facilitar a manutenção.

**Polimorfismo**

Conceito associado à herança. Realiza a redeclaração de métodos previamente herdados por uma classe.  
Esses métodos, embora semelhante, diferem de alguma forma da implementação utilizada na superclasse

|  |
| --- |
| **Polimorfismo:** métodos com o mesmo nome, mas que fazem ações parcialmente diferentes. |

Desenvolva o diagrama de classe para o cenário em que o aluno deve acessar o sistema de sua universidade para consultar notas e presença.

**Conceitos básicos para diagramas de classes**

**Atributos e métodos**

Atributos: armazenam dados de objeto da classe;  
#nro\_onta: int  
#dt\_abertura: date

Métodos: funções que podem ser executadas pelas instâncias da classe. Deve-se identificar os possíveis parâmetros recebidos e os possíveis valores retornados.  
+abrir\_Conta(): int  
+constular\_Conta(): int

|  |
| --- |
| +consultar\_Conta(int numero): int |

**Relacionamentos e associações**

Os relacionamentos, associações, permitem que as classes compartilhem informações entre si. As associações são representadas por linhas representando as classes envolvidas.  
- Associação unária ou reflexiva

**Associação unária ou reflexiva**

Ocorre quando existe o relacionamento de um objeto de uma classe com objetos da mesma classe.

**Multiplicidade**

Determina o número mínimo e máximo de objetos envolvidos em cada extremidade da associação. Adicionalmente. Especifica o nível de dependência de um objeto para com os outros envolvidos na associação.