# 1 TERMINOLOGIA BÁSICA DE ORIENTAÇÃO A OBJETOS

A terminologia de Orientação a Objetos não é utilizada de maneira uniforme na bibliografia existente sobre o assunto. Por questões de clareza, este trabalho irá adotar algumas definições e conceitos apresentados por B. Meyer.

Segundo Meyer, o projeto orientado a objetos é a construção de sistemas de *software* como coleções estruturadas de implementações de tipos abstratos de dados.

O projeto orientado a objetos propõe uma arquitetura de *software* baseada nos objetos que o sistema manipula, e não nas funções que o mesmo irá desempenhar. Portanto, um *software* orientado a objetos está muito mais relacionado com as estruturas de dados envolvidas do que com as funções.

## 1.1 Objetos

Um objeto pode ser definido como um módulo que agrupa um estado e um comportamento.

O estado interno de um objeto é representado pelos campos que são armazenados na memória primária, e pode ser alterado ao longo da vida do objeto.

O comportamento de um objeto é definido pelo seus métodos. Cada método é composto por um conjunto de instruções com a capacidade de alterar o estado do objeto. Os métodos incorporam um conjunto de ações que estão diretamente relacionadas com as estruturas de dados contidas nos campos de dados do objeto.

## 1.2 Classes

A declaração de uma classe descreve a estrutura de um conjunto ( potencialmente infinito ) de objetos. Na declaração da classe constam todos os métodos e atributos comuns para todo um conjunto de objetos que pertencem a essa classe. Todo objeto é uma instância da classe à qual ele pertence.

## 1.3 Atributos

A declaração de uma classe pode conter um ou mais atributos. Cada atributo da classe define um campo nos objetos que pertencem a essa classe. Para deixar bastante clara a diferença entre o significado dos termos atributo e campo, diz-se que uma classe possui atributos ( meras declarações ), enquanto que cada instância dessa classe possui um campo correspondente à declaração de um atributo. No exemplo abaixo, a classe **Pessoa** possui um atributo chamado **idade**, e a instância **X** possui um campo chamado **idade**.

```
class Pessoa {
        private :
            int         idade;
            char        sexo;
} X;
```

O conjunto de campos de um objeto compõe o seu estado.

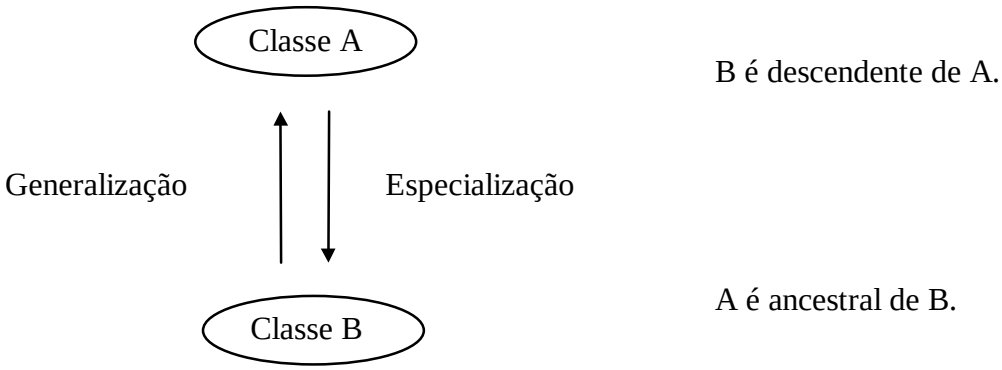
## 1.4 Métodos e mensagens

A grosso modo, os métodos são módulos executáveis que fazem parte de uma classe. Em programação orientada a objetos, a ação é iniciada por um objeto chapodar, que envia uma mensagem a um agente ( que nada mais é do que um objeto ), e que nesse instante fica responsável por uma determinada ação.

Uma mensagem contém a solicitação de uma ação, e pode seguir acompanhada de informações adicionais necessárias, os argumentos. O receptor irá cumprir a ação solicitada através da execução de um método específico.

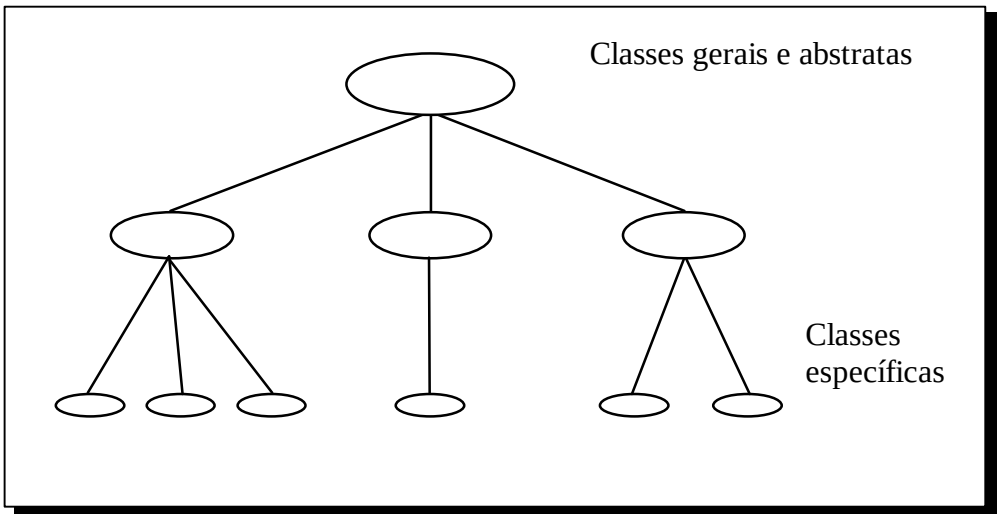
## 1.5 Herança

O mecanismo de herança permite declarar uma nova classe a partir de outra já existente. Isto pode ser feito de duas maneiras : por generalização e por especialização.




Neste caso a classe ***B*** herda automaticamente todos os atributos e métodos da classe ***A***. Na declaração da classe descendente ***B*** pode-se adicionar novos atributos e métodos. Pode-se ainda, sobrepor ou redefinir métodos e atributos herdados da classe ancestral ***A***.

O uso repetido desse mecanismo em diversos níveis de declarações de classes é chamado de **herança múltipla.**




## 1.6 Information Hiding

Pelo princípio de *Information Hiding*, somente as informações que forem realmente necessárias para a utilização de uma classe devem ser visíveis aos usuários das classes.

Todos os demais detalhes devem permanecer inacessíveis a outros objetos e a qualquer outra parte do programa.

Cada classe possui apenas uma parte visível publicamente, chamada de interface. Portanto, tudo o que não faz parte da interface de uma classe pode ser modificado ou aprimorado, sem que isto cause alguma alteração na interação do objeto com o restante do *software*.

## 1.7 Encapsulamento de dados

O encapsulamento estabelece uma relação entre os métodos e os campos de dado de um objeto. Nessa relação, o único meio de acesso aos dados de um objeto é feito através de chamadas aos métodos do mesmo objeto. Nenhuma parte do programa orientado a objetos pode operar diretamente sobre um dado do objeto. Qualquer comunicação entre os objetos de um programa ocorrerá exclusivamente por meio de mensagens explícitas.

## 1.8 Polimorfismo

Polimorfismo significa a capacidade de tomar várias formas. Em programação orientada a objetos, polimorfismo se refere à possibilidade de uma entidade referenciar instâncias de várias classes durante o tempo de execução. Geralmente isto só é possível se a entidade for declarada em uma classe ancestral, que generaliza uma ou mais classes descendentes.