Classe: onde ocorre a escrita do código, para escrever um código em java precisa ser definido uma classe antes, as classes precisam ser instanciadas na classe principal para poderem ser utilizadas no código, as classes serão utilizadas em conjunto para gerar um código orientado a objeto.

Objeto: a base do que deve ser pensado para criar um código orientado a objeto, o objeto deve possuir métodos e atributos que façam sentido e condizem com a função do objeto na vida real e/ou no código implementado, o objeto deve descrever suas próprias funcionalidades e ações.

Métodos e Atributos: componentes do objeto, os atributos funcionam como características do objeto, definindo o objeto, já os métodos são as funcionalidades dos objetos, geralmente utilizam os atributos para definir o que o objeto é capaz de fazer, tanto métodos como objetos precisam ser chamados na classe principal para serem utilizados, porém em algumas situações os métodos retornam o valor dos atributos.

Encapsulamento: é um dos princípios fundamentais da programação orientada a objetos, consiste em esconder os métodos e/ou atributos de um objeto com o objetivo de eles não serem acessados por outras classes, geralmente isso é feito para ter controle sobre o código, e é usado na maioria das vezes o comando private que será falado mais no próximo tópico.

Visibilidade de atributos e métodos: no início do estudo de orientação a objeto começamos utilizando métodos e atributos que são totalmente visíveis, para isso é utilizado o comando public antes da definição dos componentes, já o comando protected permite que o componente seja acessado apenas por outras classes do mesmo pacote, já o comando private serve para esconder completamente o método ou atributo e só poderão ser acessados em outras classes com comandos Get e Set definidos na classe do objeto, o comando set pode ser usado também para definir uma padronagem na definição de um atributo.

Herança: método essencial usado na construção de um código orientado a objeto, ele consiste em fazer um objeto herdar o outro, ou seja, ele será o filho de outro elemento, possuirá os mesmos atributos e métodos, a unica regra é que o filho

não pode ter menos que o pai (objeto usado para herdar), mas o filho pode possuir mais métodos que o pai e pode ser herdado em outro objeto, o filho se torna um tipo de pai, pode ser instanciado no lugar onde o pai poderia.

Polimorfismo: é um conceito importante da programação orientada a objetos que permite que objetos de diferentes classes sejam tratados de maneira igual, segue do conceito de herança, ou seja uma classe herda da outra e classe usada como pai vira polimorfa pois a herdada tambem será considerada do mesmo tipo que a classe pai.

Classes abstratas: as classes abstratas são usadas apenas para servirem como pai, outras classes herdam dela e elas não podem ser instanciadas na classe principal, geralmente são usadas para que uma classe se torne polimorfa mas o elemento principal não é considerado uma forma útil, ex: Cliente(classe abstrata) tipos de cliente(filhos da classe cliente).

Construtores: são métodos chamados quando se instancia uma classe, podem ser usados para obrigar uma classe a executar um determinado método e/ou para definir atributos de um objeto, um objeto pode possuir diversor contrutores com cada um recebendo elementos diferentes.

Get e Set: são usados para resgatar atributos do tipo privado, o método get exibe o atributo em outra classe e o set define um valor para o atributo, eles só podem ser usados se forem montados na classe do objeto, podem ser usados para definir regras na montagem de atributos também.

Sobrecarga de métodos: consiste na criação de vários métodos com o mesmo nome mas que retornam elementos de tipo diferente e/ou recebem atributos diferentes em sua construção, a sobrecarga de métodos pode ser usada também no construtor.

Sobrescrita de métodos: consiste na reescrita de um métodos herdado de uma classe pai(herança), com isso a filha executará uma ação diferente da classe pai mesmo com o mesmo método.

Palavras reservadas (super, this e final): a palavra super serve para resgatar os

atributos herdados de uma classe pai no construtor de uma classe filha, a palavra this serve para se referir a um atributo que foi gerado na classe, também usada geralmente no construtor já que para a definição das variáveis geralmente o construtor recebe como parâmetros elementos com nomes iguais os das classes, precisando então serem especificados,a palavra final serve para demonstrar que determinado elemento no código não pode ser alterado, pode ser usada em classes, métodos e atributos.

Relação de objetos: “ter”, “usar” e “ser”: "ter" seria os atributos de um objeto, o objeto tem determinados atributos, "usar" seria os métodos de um objeto, ele usa o método para realizar determinadas funcionalidades, "ser" seria definido a partir de uma relação de polimorfismo, exemplo: uma classe Fiat herda de uma classe carro no caso fiat "seria" um carro.