**Equals e HashCode**

Quando a gente cria objetos na memoria eles possuem atributos, então a gente pega la por exemplo pessoa física, pessoa física vai ter la o nome completo, cpf, data de nascimento, todos esses são atributos e é muito comum a gente precisar comparar objetos, eu quero saber se um determinado objeto é igual ao o outro.

Como que eu sei que dois objetos são iguais ? A forma pra descobrir se dois objetos são iguais a gente precisa comparar os atributos desses dois objetos, se os atributos desses dois objetos forem iguais, logos esses objetos são iguais, apesar de serem instancias diferentes possuem os mesmos atributos por isso são iguais.

Pra isso, pra a gente trabalhar com equals que quer dizer igual ou não a gente precisa de um mecanismo que a gente chama de a geração de uma identificação única desse objeto, como se a gente tivesse um cpf desse objeto.

Se a gente olhar a classe Object, ela tem dois métodos implementados equals() e hashcode(), ela tem o método **equals()** que retorna TRUE se ambas variáveis apontam para o mesmo objeto e FALSE caso contrario, o que que acontece é o seguinte, Object não tem atributos, porque é uma classe pai de todas as classes do java, então toda vez que a gente precisar comparar dois objetos a gente vai precisar implementar o equals() pra permitir a comparação igual entre dois objetos, então nos temos o método equals que é um método boolean que retorna true ou false.

E temos o método chamado **hashcode() 🡪** Traduzindo pro português brasileiro: Código de dispersão ou código hash: é um numero inteiro que é gerado para representar/identificar um objeto durante a execução do programa, então é um numero gerado usando uma formula padrão pra gerar um numero único que vai identificar esse objeto durante a execução do programa.

Todos os objetos eles precisam ter esses dois métodos implementados, justamente pra a gente conseguir comparar se eles são iguais e pra cada objeto ter o seu numero de identificação única, então por exemplo objeto com os mesmos atributos vão ter o mesmo hashcode, mas mesmo assim irão ter referencias de memoria diferente, durante a mesma execução, então se a gente fizer uma comparação de pessoas físicas com o mesmo nome, cpf e data de nascimento a gente vai ter o mesmo hashcode

Regras de transitividade:

Quando eu faço: x.equals(x) == TRUE 🡪 x.equals e o mesmo objeto(x) vai retornar true, porque eles estão apontando para a mesma referencia/endereço de memoria, pois esta comparando com o mesmo objeto, então é o mesmo objeto

x.equals(y) == TRUE -> y.equals(x) == TRUE 🡪 Se x.equals(y) for true, então y.equals(x) também vai ser true, é a questão da inversão da formula, então se eu inverter os operandos eu vou ter o mesmo resultado

x.equals(y) == TRUE e y.equals(z) == TRUE -> x.equals(z) == TRUE 🡪 mesma coisa quando eu tenho três variáveis, se X é igual a y e y é igual a z, então eu posso falar que x também é igual a z, transitividade

x.equals(y) == TRUE -> x.hashcode() == y.hashcode() 🡪 se x é igual a y, então quer dizer que o hascode de x é igual ao hashcode de y, então o numero de identificação dos dois objetos é igual

**PRATICAS NA PASTA EqualsEHascode tudo documentado la**