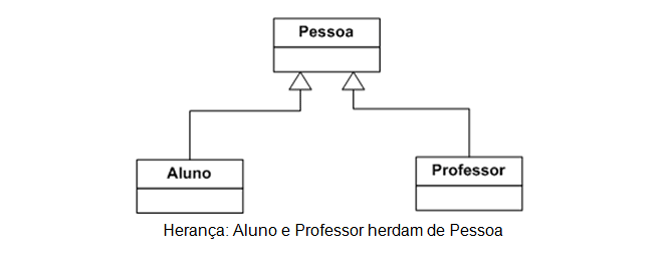
Review Classe

Uma **Classe**"**class**" fornecem um meio de juntar dados e funcionalidades/métodos, ou seja uma classe contém um conjunto de objetos/métodos relacionados. Criar uma nova classe cria um novo tipo de objeto, permitindo que novas instâncias desse tipo sejam feitas. Cada instância de classe pode ter atributos anexados a ele para manter seu estado. As instâncias de classe também podem ter métodos (definidos por sua classe) para modificar seu estado.

 As classes em Python fornecem todos os recursos padrão da Programação Orientada a Objetos, em python é permitido herança múltiplas de classes, ou seja uma classe pode herdar por padrão outras classes sem problemas, em java e ruby não é permitido uma classe herdar direto outras, existe outras formas pra fazer isso,que em python se faz com facilidade.

Os objetos podem conter quantidades arbitrárias e tipos de dados. Como é verdade para os módulos, as classes participam da natureza dinâmica do Python: são criadas no tempo de execução e podem ser modificadas após a criação.



A maior vantagem de usar classe é porque você pode herdar funcionalidade comuns em várias outras classes, com isso podemos reutilizar nosso código, como no exemplo acima os métodos relacionados a Pessoa é herdado por aluno e professor. Geralmente uso classes pra Herdar os métodos genéricos criados na minha classe mãe.

No nosso curso utilizamos o básico de classes, para não ficar muito complicado para quem está iniciando automação de testes.

Existem pelo menos duas formas diferentes de trabalhar com os parâmetros de entrada. Neste exemplo, definiremos o **parâmetro apenas uma vez** com um método especial do Python chamado \_\_init\_\_. Este método é chamado quando um objeto de uma classe é instanciado. Este método é útil para fazer qualquer inicialização que você queira com seu objeto, ou seja, ele é o método **"Inicializador"** da instancia.

1. # Criamos uma classe com a palavra reservada "class", em seguida o nome da classe, e ":" no final

4. class NomeDaClasse:
5. propriedade = 0 # variavel da classe
7. def \_\_init\_\_(self, valor):
8. self.variavel\_de\_instancia = valor
10. def metodo(self):
11. variavel\_local = 1000 # somente enxergo dentro do metodo
12. print(self.propriedade) # se tiver só propriedade aqui seria uma variavel local
13. print(self.variavel\_de\_instancia)
14. print(variavel\_local)

17. # Realiza a Instancia a classe, para ter acesso aos métodos.
18. # Atraves \_\_init\_\_ passamos o valor direto na instancia da classe.
20. objeto = NomeDaClasse(100)
21. objeto.propriedade = 10 # Mudando o valor da variavel da classe.
22. objeto.metodo() # Realizando acesso a funcao/metodo da classe.
24. # resultado: 10, 100, 1000

Através da palavra reservada "self", você pode chamar a variável da classe e a variável de instancia, dentro de qualquer método/função.

Referência:

https://docs.python.org/3.6/tutorial/classes.html