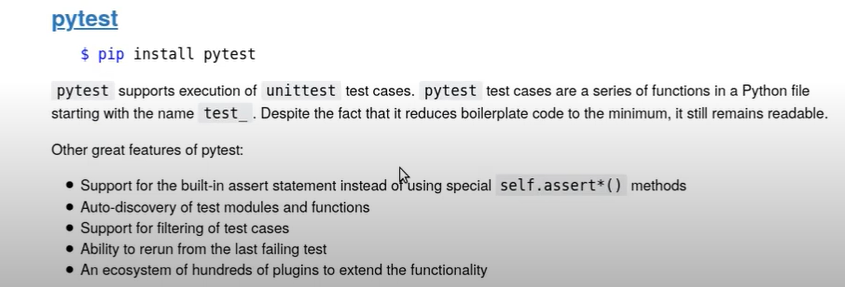
**Unit Testing in Python using PyTest**

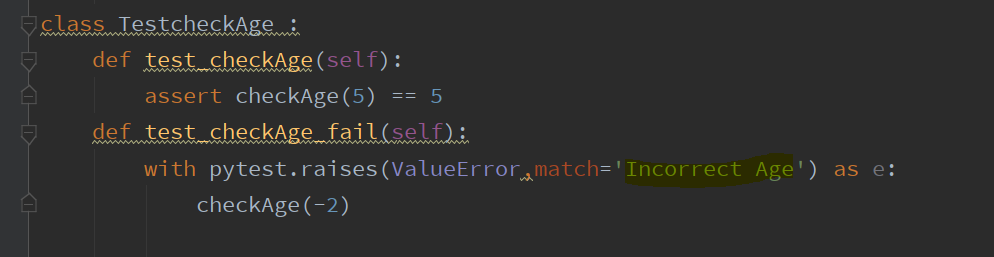


* While you write unit test cases using pytest:
  + Name of the Test case python file name should start with ***test***
  + Neame Functions which are your test cases should start with ***test***
  + Name of the Classes with multiple test case functions (test set) should start with ***Test***
  + On Failure assert raises ‘AssertionError’’
  + While writing negative test cases to make the function pass , you can use(Catching Expected Exceptions :

with pytest.raises(‘AssertionError’):

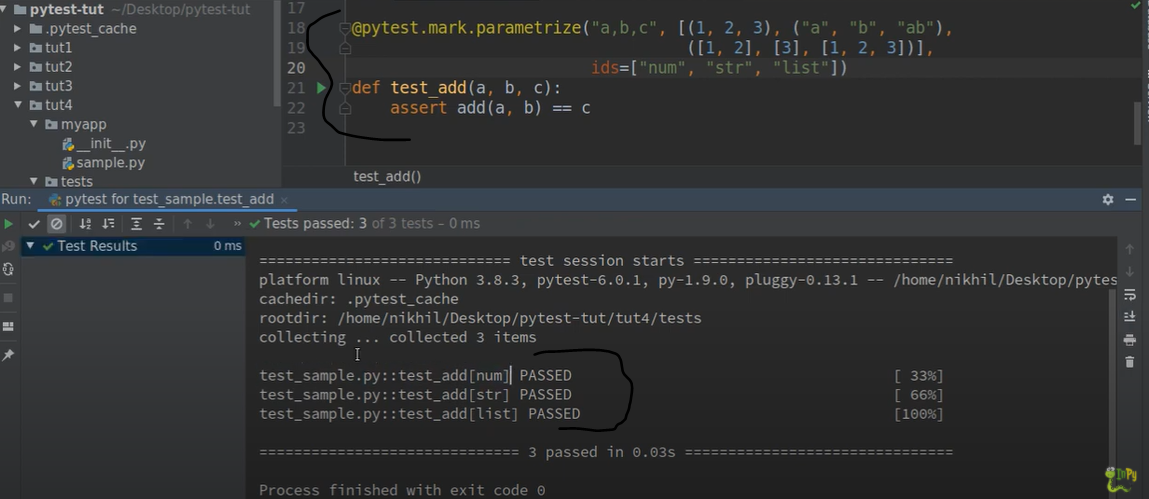
Assert func(\*args,\*\*kwargs) == Expected\_Result

* + To Catch Custom exceptions or any built in exceptions and validate also , you can use the above pytest.raises()
  + pytest.raises returns an exceptionInfo object and the value can be accessed through .value attribute
  + Raised messages can be validated as below

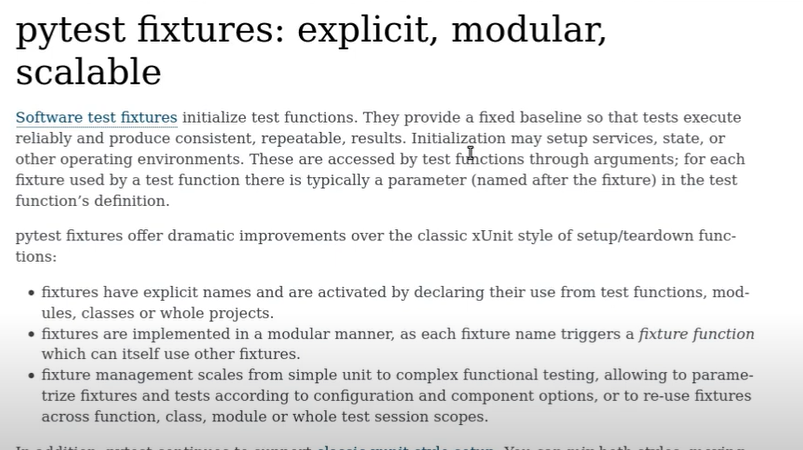


**Markers**

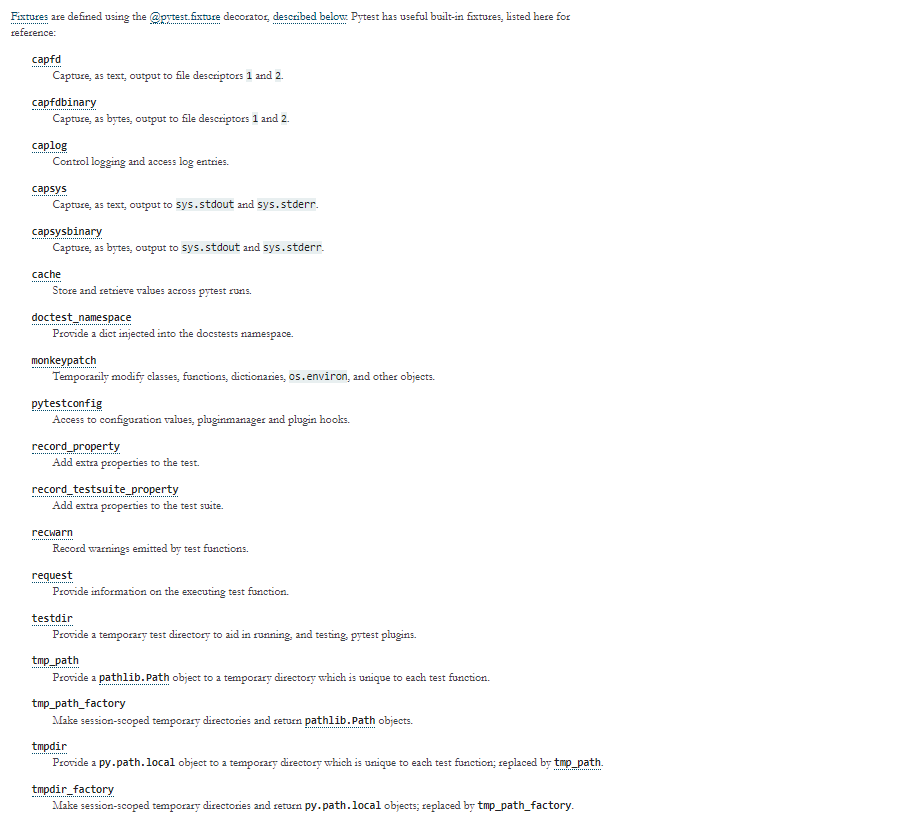
* If you want to skip a test case (function) or a test set (class) , we can use ***@pytest.mark.skip***
* Conditional skip ***@pytest.mark.skipif()***
* To handle expected failures ***@pytest.mark.xfail()***
* To Parameterize the function with ***@pytest.mark.parametrize()***



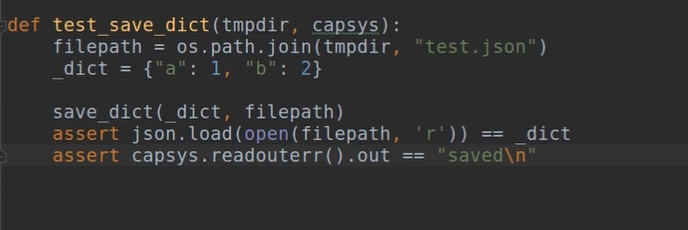
* Pytest Fixtures:



**Built in Fixtures:**



Example of using built in fixtures:



**Custom Pytest Fixtures**

