Prac10

testing.py

*"""  
CP1404/CP5632 Practical  
Testing demo using assert and doctest  
"""*import doctest  
from car import Car  
  
  
def repeat\_string(s, n):  
 *"""Repeat string s, n times, with spaces in between."""* str=**" "** return str.join(s \* n)  
  
  
def is\_long\_word(word, length=5):  
 *"""  
 Determine if the word is as long or longer than the length passed in  
 >>> is\_long\_word("not")  
 False  
 >>> is\_long\_word("supercalifrag")  
 True  
 >>> is\_long\_word("Python", 6)  
 True  
 """* return len(word) >= length  
  
  
def run\_tests():  
 *"""Run the tests on the functions."""  
 # assert test with no message - used to see if the function works properly* print(repeat\_string(**"hi"**, 2))  
 assert repeat\_string(**"Python"**, 1) == **"Python"** *# the test below should fail* assert repeat\_string(**"hi"**, 2) == **"hi hi"** *# TODO: 1. fix the repeat\_string function above so that it passes the failing test  
 # Hint: "-".join(["yo", "yo"] -> "yo-yo"  
  
 # assert test with custom message,  
 # used to see if Car's init method sets the odometer correctly  
 # this should pass (no output)* test\_car = Car()  
 assert test\_car.odometer == 0, **"Car does not set odometer correctly"** *# TODO: 2. write assert statements to show if Car sets the fuel correctly  
 # Note that Car's \_\_init\_\_ function sets the fuel in one of two ways:  
 # using the value passed in or the default  
 # You should test both of these* test\_car = Car(fuel=10)  
 assert test\_car.fuel == 10  
  
  
run\_tests()  
  
*# TODO: 3. Uncomment the following line and run the doctests  
# (PyCharm may see your >>> doctest comments and run doctests anyway.)*doctest.testmod()  
  
*# TODO: 4. Fix the failing is\_long\_word function  
# (don't change the tests, change the function!)  
  
# TODO: 5. Write and test a function to format a phrase as a sentence,  
# starting with a capital and ending with a single full stop.  
# Important: start with a function header and just use pass as the body  
# then add doctests for 3 tests:  
# 'hello' -> 'Hello.'  
# 'It is an ex parrot.' -> 'It is an ex parrot.'  
# and one more you decide (one that is valid!)  
# test this and watch the tests fail  
# then write the body of the function so that the tests pass*def format\_sentence(phrase):  
 sentence = phrase.capitalize()+**'.'** return sentence  
  
def text\_sentence():  
 phrase=str(input(**'Enter a phrase: '**))  
 sentence = format\_sentence(phrase)  
 print(sentence)  
  
text\_sentence()

wiki.py

import wikipedia  
  
text = str(input(**'Search: '**))  
wikipedia.search(text, results=10)  
  
try:  
 wikipedia.summary(text)  
except wikipedia.exceptions.DisambiguationError as e:  
 print(e.options)  
  
text\_page = wikipedia.page(text)  
print(**'Title: {}'**.format(text\_page.title))  
print(**'URL: {}'**.format(text\_page.url))