

```

1 #DS5100 Module 08 Homework
2 #ID : VVK6RD
3 #Name: Ade Faparusi
4
5 import pandas as pd
6
7 class BookLover():
8     """
9     Just another Python class.
10    """
11
12
13    def __init__(self, name, email, fav_genre):
14        self.name = name
15        self.email = email
16        self.fav_genre = fav_genre
17        self.num_books = 0
18        self.book_list = pd.DataFrame({'book_name':[], 'book_rating':[]})
19
20
21    def add_book(self, book_name, rating):
22        self.book_name = book_name
23        self.rating = rating
24
25        if (len(self.book_list) > 0) and (self.book_name in self.book_list.book_name.values):
26            print(self.book_name, " already on book list")
27            return False
28        else:
29            new_book = pd.DataFrame({
30                'book_name': [self.book_name],
31                'book_rating': [self.rating]
32            })
33            self.book_list = pd.concat([self.book_list, new_book], ignore_index=True)
34
35
36    def has_read(self, book_name):
37        self.book_name = book_name
38        if self.book_name in self.book_list.book_name.values:
39            print ("You have read the book: ", self.book_name)
40            return True
41        else:
42            print ("You have not read the book: ", self.book_name)
43            return False
44
45
46    def num_books_read(self):
47        actual = len(self.book_list)
48        print ('num_books_read: ', actual)
49        return actual
50
51    def fav_books(self):
52        fav_list = [num for num in self.book_list[self.book_list.book_rating>3].book_name.values]
53        print ("List of favorite books : ",fav_list)
54        return fav_list
55
56
57 if __name__ == '__main__':
58
59     test_object = BookLover("Han Solo", "hsolo@millenniumfalcon.com", "scifi")
60     test_object.add_book("War of the Worlds", 4)
61     test_object.add_book("War of the Worlds", 4)
62     test_object.add_book("Jane Eyre", 4)
63     test_object.add_book("Fight Club", 3)
64     test_object.add_book("The Divine Comedy", 5)
65     test_object.add_book("The Popol Vuh", 5)
66     test_object.has_read('The Popol Vuh')

```

```

67     test_object.has_read('The Roll')
68     test_object.num_books_read()
69     test_object.fav_books()
70
71
72
73
74
75
76
77
78
79
80     #DS5100 Module 08 Homework
81     #ID : VVK6RD
82     #Name: Ade Faparusi
83
84
85     import unittest
86     from booklover import BookLover
87
88
89     class BookLoverTestSuite(unittest.TestCase):
90
91         def test_1_add_book(self):
92             # add a book and test if it is in 'book_list'.
93             booklover1 = BookLover("Han Solo", "hsolo@millenniumfalcon.com", "scifi")
94             test_book_name = "Oliver Twist"
95             test_rating = 4
96             booklover1.add_book(test_book_name, test_rating)
97             self.assertTrue(booklover1.has_read(test_book_name))
98
99
100        def test_2_add_book(self):
101            # add the same book twice. Test if it's in 'book_list' only once.
102            booklover1 = BookLover("Han Solo", "hsolo@millenniumfalcon.com", "scifi")
103            test_book_name = "Oliver Twist"
104            test_rating = 4
105            booklover1.add_book(test_book_name, test_rating)
106            booklover1.add_book(test_book_name, test_rating)
107            count_book = len (booklover1.book_list[booklover1.book_list.book_name ==test_book_name])
108            self.assertEqual(count_book, 1)
109
110        def test_3_has_read(self):
111            # pass a book in the list and test if the answer is 'True'.
112            booklover1 = BookLover("Han Solo", "hsolo@millenniumfalcon.com", "scifi")
113            test_book_name = "Oliver Twist"
114            test_rating = 4
115            booklover1.add_book(test_book_name, test_rating)
116            self.assertTrue(booklover1.has_read(test_book_name))
117
118        def test_4_has_read(self):
119            # pass a book NOT in the list and use 'assert False' to test the answer is 'True'
120            booklover1 = BookLover("Han Solo", "hsolo@millenniumfalcon.com", "scifi")
121            test_book_name = "Oliver Twist"
122            test_rating = 4
123            booklover1.add_book(test_book_name, test_rating)
124            test_book_name2 = "Monte Cristo"
125            self.assertFalse(booklover1.has_read(test_book_name2))
126
127        def test_5_num_books_read(self):
128            # add some books to the list, and test num_books matches expected.
129            booklover1 = BookLover("Han Solo", "hsolo@millenniumfalcon.com", "scifi")
130            add_books = [("Oliver Twist",4),
131                         ("Jane Eyre", 4),
132                         ("Fight Club", 3),

```

```

133         ("The Divine Comedy", 5),
134         ("The Popol Vuh", 5)
135     ]
136     for book_info in add_books:
137         booklover1.add_book(*book_info)
138
139     self.assertEqual(booklover1.num_books_read(), len(add_books))
140
141
142     def test_6_fav_books(self):
143         # add some books with ratings to the list, making sure some of them have rating > 3.
144         # Your test should check that the returned books have rating > 3
145         booklover1 = BookLover("Han Solo", "hsolo@millenniumfalcon.com", "scifi")
146         add_books = [("Oliver Twist", 4),
147                     ("Jane Eyre", 4),
148                     ("Fight Club", 3),
149                     ("The Divine Comedy", 5),
150                     ("The Popol Vuh", 5)
151                     ]
152         for book_info in add_books:
153             booklover1.add_book(*book_info)
154
155         self.assertEqual(len(booklover1.fav_books()), len([a for a, b in add_books if b > 3]))
156
157     if __name__ == '__main__':
158
159         unittest.main(verbosity=3)
160
161     test_1_add_book (__main__.BookLoverTestSuite) ... ok
162     test_2_add_book (__main__.BookLoverTestSuite) ... ok
163     test_3_has_read (__main__.BookLoverTestSuite) ... ok
164     test_4_has_read (__main__.BookLoverTestSuite) ... ok
165     test_5_num_books_read (__main__.BookLoverTestSuite) ... ok
166     test_6_fav_books (__main__.BookLoverTestSuite) ... ok
167
168     -----
169     Ran 6 tests in 0.030s
170
171     OK

```

```

1 #DS5100 Module 08 Homework
2 #ID : VVK6RD
3 #Name: Ade Faparusi
4
5 import pandas as pd
6
7 class BookLover():
8     """
9     Just another Python class.
10    """
11
12
13    def __init__(self, name, email, fav_genre):
14        self.name = name
15        self.email = email
16        self.fav_genre = fav_genre
17        self.num_books = 0
18        self.book_list = pd.DataFrame({'book_name':[], 'book_rating':[]})
19
20
21    def add_book(self, book_name, rating):
22        self.book_name = book_name
23        self.rating = rating
24
25        if (len(self.book_list) > 0) and (self.book_name in self.book_list.book_name.values):
26            print(self.book_name, " already on book list")
27            return False
28        else:
29            new_book = pd.DataFrame({
30                'book_name': [self.book_name],
31                'book_rating': [self.rating]
32            })
33            self.book_list = pd.concat([self.book_list, new_book], ignore_index=True)
34
35
36    def has_read(self, book_name):
37        self.book_name = book_name
38        if self.book_name in self.book_list.book_name.values:
39            print ("You have read the book: ", self.book_name)
40            return True
41        else:
42            print ("You have not read the book: ", self.book_name)
43            return False
44
45
46    def num_books_read(self):
47        actual = len(self.book_list)
48        print ('num_books_read: ', actual)
49        return actual
50
51    def fav_books(self):
52        fav_list = [num for num in self.book_list[self.book_list.book_rating>3].book_name.values]
53        print ("List of favorite books : ",fav_list)
54        return fav_list
55
56
57 if __name__ == '__main__':
58
59     test_object = BookLover("Han Solo", "hsolo@millenniumfalcon.com", "scifi")
60     test_object.add_book("War of the Worlds", 4)
61     test_object.add_book("War of the Worlds", 4)
62     test_object.add_book("Jane Eyre", 4)
63     test_object.add_book("Fight Club", 3)
64     test_object.add_book("The Divine Comedy", 5)
65     test_object.add_book("The Popol Vuh", 5)
66     test_object.has_read('The Popol Vuh')

```

```

67     test_object.has_read('The Roll')
68     test_object.num_books_read()
69     test_object.fav_books()
70
71
72
73
74
75
76
77
78
79
80     #DS5100 Module 08 Homework
81     #ID : VVK6RD
82     #Name: Ade Faparusi
83
84
85     import unittest
86     from booklover import BookLover
87
88
89     class BookLoverTestSuite(unittest.TestCase):
90
91         def test_1_add_book(self):
92             # add a book and test if it is in 'book_list'.
93             booklover1 = BookLover("Han Solo", "hsolo@millenniumfalcon.com", "scifi")
94             test_book_name = "Oliver Twist"
95             test_rating = 4
96             booklover1.add_book(test_book_name, test_rating)
97             self.assertTrue(booklover1.has_read(test_book_name))
98
99
100        def test_2_add_book(self):
101            # add the same book twice. Test if it's in 'book_list' only once.
102            booklover1 = BookLover("Han Solo", "hsolo@millenniumfalcon.com", "scifi")
103            test_book_name = "Oliver Twist"
104            test_rating = 4
105            booklover1.add_book(test_book_name, test_rating)
106            booklover1.add_book(test_book_name, test_rating)
107            count_book = len (booklover1.book_list[booklover1.book_list.book_name ==test_book_name])
108            self.assertEqual(count_book, 1)
109
110        def test_3_has_read(self):
111            # pass a book in the list and test if the answer is 'True'.
112            booklover1 = BookLover("Han Solo", "hsolo@millenniumfalcon.com", "scifi")
113            test_book_name = "Oliver Twist"
114            test_rating = 4
115            booklover1.add_book(test_book_name, test_rating)
116            self.assertTrue(booklover1.has_read(test_book_name))
117
118        def test_4_has_read(self):
119            # pass a book NOT in the list and use 'assert False' to test the answer is 'True'
120            booklover1 = BookLover("Han Solo", "hsolo@millenniumfalcon.com", "scifi")
121            test_book_name = "Oliver Twist"
122            test_rating = 4
123            booklover1.add_book(test_book_name, test_rating)
124            test_book_name2 = "Monte Cristo"
125            self.assertFalse(booklover1.has_read(test_book_name2))
126
127        def test_5_num_books_read(self):
128            # add some books to the list, and test num_books matches expected.
129            booklover1 = BookLover("Han Solo", "hsolo@millenniumfalcon.com", "scifi")
130            add_books = [("Oliver Twist",4),
131                        ("Jane Eyre", 4),
132                        ("Fight Club", 3),

```

```

133         ("The Divine Comedy", 5),
134         ("The Popol Vuh", 5)
135     ]
136     for book_info in add_books:
137         booklover1.add_book(*book_info)
138
139     self.assertEqual(booklover1.num_books_read(),len(add_books))
140
141
142     def test_6_fav_books(self):
143         # add some books with ratings to the list, making sure some of them have rating > 3.
144         # Your test should check that the returned books have rating > 3
145         booklover1 = BookLover("Han Solo", "hsolo@millenniumfalcon.com", "scifi")
146         add_books = [("Oliver Twist",4),
147                     ("Jane Eyre", 4),
148                     ("Fight Club", 3),
149                     ("The Divine Comedy", 5),
150                     ("The Popol Vuh", 5)
151                 ]
152         for book_info in add_books:
153             booklover1.add_book(*book_info)
154
155         self.assertEqual(len(booklover1.fav_books()),len([a for a,b in add_books if b>3]))
156
157     if __name__ == '__main__':
158
159         unittest.main(verbosity=3)
160
161     test_1_add_book (__main__.BookLoverTestSuite) ... ok
162     test_2_add_book (__main__.BookLoverTestSuite) ... ok
163     test_3_has_read (__main__.BookLoverTestSuite) ... ok
164     test_4_has_read (__main__.BookLoverTestSuite) ... ok
165     test_5_num_books_read (__main__.BookLoverTestSuite) ... ok
166     test_6_fav_books (__main__.BookLoverTestSuite) ... ok
167
168     -----
169     Ran 6 tests in 0.030s
170
171     OK

```