테스트 보고서

컴퓨터공학부 20153240 한채연

테스트 방법 : assertTrue() 혹은 assertFalse() 메소드를 이용하여 테스트 진행

testDataAdmin.py (class TestDataAdmin(unittest.TestCase))

1. testAddClick : 데이터 추가 test

```
def testAddClick(self):
    self.assertTrue(self.add_click({1: ["Marge", 40], 2: ["Bart", 77], 3: ["Hommer", 25], 4: ["Lisa", 100]},"\tsimpson", "44"))
    self.assertTrue(self.add_click({1: ["Karl", 55], 2: ["Rick", 23], 3: ["Laury", 99]}, "Tom", "25")) #2
    self.assertFalse(self.add_click({1: ["Marge", 40], 2: ["Lisa", 40]}, " ", "34")) #3
    self.assertFalse(self.add_click({1: ["Karl", 55], 2: ["Rick", 23]}, "Amy", 'abc')) #4
    self.assertFalse(self.add_click({1: ["Karl", 55], 2: ["Amy", 23]}, "Karl", "89")) #5
```

#1 : tab 넘김이 되어있는 이름이 넘어갈 때에 tab 넘김을 제거하고 정상적으로 데이터가 저장되는지 확인

#2 : 모두 정상적인 데이터

#3 : 이름이 공란일 때에 올바른 error 처리가 되는지 확인

#4 : 점수 타입이 올바르지 않을 때 error 처리가 되는지 확인

#5 : 동일한 이름이 존재할 때 올바른 error 처리가 되는지 확인

<테스트 결과 : 터미널에서 확인>

```
hanchaeyeon-ui-MacBook-Air:untitled1 Chaeyeon$ python -m unittest -v testDataAdm
in
testAddClick (testDataAdmin.TestDataAdmin) ... 성공적으로 추가하였습니다
성공적으로 추가하였습니다
[추가 실패] 이름을 다시 입력해 주세요.
[추가 실패] 점수를 다시 입력하세요.
[추가 실패] 동일한 이름이 존재합니다.
ok
```

2. testDelClick : 데이터 삭제 test

```
def testDelClick(self):
    self.assertTrue(self.del_click({1: ["Marge", 40], 2: ["Bart", 77], 3: ["Hommer", 25], 4: ["Lisa", 100]},"2")) #1
    self.assertFalse(self.del_click({1: ["Karl", 55], 2: ["Rick", 23], 3: ["Laury", 99]}, "5")) #2
    self.assertFalse(self.del_click({4:["chae",100], 2: ["yeon",44]},"a")) #3
```

#1 : 모두 정상적인 데이터

#2 : index 값의 범위가 벗어날 때 올바른 error 처리가 되는지 확인

#3 : index 타입이 올바르지 않을 때 올바른 error 처리가 되는지 확인

<테스트 결과>

```
testDelClick (testDataAdmin.TestDataAdmin) ... 성공적으로 삭제하였습니다
삭제에 실패했습니다.
삭제에 실패했습니다.
ok
```

3. data sorting test

```
def testSortIdx(self):
    self.assertTrue(self.sort_idx({5: ["Marge", 40], 3 :  ["Bart", 77], 1: ["Hommer", 25], 2: ["Lisa", 100]})) #1

def testSortName(self):
    self.assertTrue(self.sort_name({1: ["Marge", 40], 2: ["Bart", 77], 3: ["Hommer", 25], 4: ["Lisa", 100]})) #2

def testSortScoreRev(self):
    self.assertTrue(self.sort_scorerev({1: ["Marge", 40], 2: ["Bart", 77], 3: ["Hommer", 25], 4: ["Lisa", 100]})) #3

def testSortScore(self):
    self.assertTrue(self.sort_score({1: ["Marge", 40], 2: ["Bart", 77], 3: ["Hommer", 25], 4: ["Lisa", 100]})) #4
```

#1 : index 순서가 섞여 있을 때 번호순으로 올바르게 sorting 되는지 확인

#2 : name 순서가 섞여 있을 때 이름순으로 올바르게 sorting 되는지 확인

#3 : 점수 내림차순으로 올바르게 sorting 되는지 확인

#4 : 점수 오름차순으로 올바르게 sorting 되는지 확인

<테스트 결과>

```
testSortIdx (testDataAdmin.TestDataAdmin) ... [(1, ['Hommer', 25]), (2, ['Lisa',
 100]), (3, ['Bart', 77]), (5, ['Marge', 40])]
ok
testSortName (testDataAdmin.TestDataAdmin) ... [(2, ['Bart', 77]), (3, ['Hommer'
, 25]), (4, ['Lisa', 100]), (1, ['Marge', 40])]
ok
testSortScore (testDataAdmin.TestDataAdmin) ... [(3, ['Hommer', 25]), (1, ['Marg
e', 40]), (2, ['Bart', 77]), (4, ['Lisa', 100])]
ok
testSortScoreRev (testDataAdmin.TestDataAdmin) ... [(4, ['Lisa', 100]), (2, ['Ba
rt', 77]), (1, ['Marge', 40]), (3, ['Hommer', 25])]
ok

----------------------------------------------------------------------
Ran 6 tests in 0.001s

OK
```

testFileAdmin.py (class TestFileAdmin(unittest.TestCase))

1. testSave : 파일 저장 test

```
def testSave(self):
    self.assertTrue(self.save_file({1: ["Marge", 40], 2: ["Bart", 77], 3: ["Hommer", 25], 4: ["Lisa", 100]}, "The Simpson")) #1
    self.assertTrue(self.save_file({1:["Karl",55], 2:["Rick", 23], 3:["Laury",99]},"\tWalking Dead\t")) #2
    self.assertFalse(self.save_file({1: ["Karl", 55], 2: ["Rick", 23], 3: ["Laury", 99]}, "\t")) #3
```

#1 : 모두 정상적인 데이터

#2 : 파일 이름에 tab 넘김이 있을 경우 tab 넘김 제거 후 정상적으로 파일 저장이 되는지 확인
#3 : 파일 이름에 tab 넘김만 있을 경우 tab 넘김 제거 후 공란으로 인식하여 정상적인 error 처리를 하는지 확인

<테스트 결과>

```
testSave (testFileAdmin.TestFileAdmin) ... 파일 저장에 성공했습니다.
파일 저장에 성공했습니다.
파일 저장에 실패했습니다.
ok

----------------------------------------------------------------------
Ran 2 tests in 0.001s
```

2. testOpen : 파일 열기 test

```
def testOpen(self):
    self.assertTrue(self.open_file("\tThe Simpson\t")) #1
    self.assertFalse(self.open_file("Wow")) #2
    self.assertFalse(self.open_file("")) #3
```

#1 : tab 넘김이 있는 이름을 tab 넘김 제거 후 정상적으로 파일을 읽을 수 있는지 확인
#2 : 입력된 이름의 파일이 존재하지 않을 경우 정상적인 error 처리를 하는지 확인
#3 : 입력된 파일 이름이 공란일 경우 정상적인 error 처리를 하는지 확인

<테스트 결과>

```
testOpen (testFileAdmin.TestFileAdmin) ... 성공적으로 파일을 읽었습니다. (파일
이름 : The Simpson)
{1: ['Marge', 40], 2: ['Bart', 77], 3: ['Hommer', 25], 4: ['Lisa', 100]}
파일 불러오기에 실패했습니다.
파일 불러오기에 실패했습니다.
ok
```