Python Class Exercise Set 7

Files

Exercise

```
fhr = open('./imdb.csv', 'r', encoding='utf-8', errors='ignore')

for line in fhr:
    print (line)

fhr.close()
```

Exercise

```
fhr = open('./imdb.csv', 'r', encoding='utf-8', errors='ignore')

for line in fhr:
    ls = line.split(',')
    print (ls)

fhr.close()
```

Exercise

```
fhr = open('./imdb.csv', 'r', encoding='utf-8', errors='ignore')

for line in fhr:
    ls = line.split(',')
    title = (ls[11]).strip()
    print (title)

fhr.close()
```

Exercise

```
fhr = open('./imdb.csv', 'r', encoding='utf-8', errors='ignore')

for line in fhr:
    ls = line.split(',')
    title = (ls[11]).strip()
    rating = (ls[25]).strip()
    print (title, rating)

fhr.close()
```

Exercise

```
fhr = open('./imdb.csv', 'r', encoding='utf-8', errors='ignore')

for line in fhr:
    ls = line.split(',')
    director = (ls[1]).strip()
    title = (ls[11]).strip()
    rating= (ls[25]).strip()
    print (director, title, rating)

fhr.close()
```

Exercise

```
fhr = open('./imdb.csv', 'r', encoding='utf-8', errors='ignore')

fhr.readline()
for line in fhr:
    ls = line.split(',')
    director = (ls[1]).strip()
    title = (ls[11]).strip()
    rating= (ls[25]).strip()
    print (director, title, rating)

fhr.close()
```

Exercise

```
fhr = open('./imdb.csv', 'r', encoding='utf-8', errors='ignore')

# {title:rating}
# {'The Ice Storm': '7.5', 'Unforgiven': '8.3'}
imdbD = {}
fhr.readline()
for line in fhr:
    ls = line.split(',')
    title = (ls[11]).strip()
    rating = (ls[25]).strip()
    if (title not in imdbD.keys()):
        imdbD[title] = rating
print (imdbD)

fhr.close()
```

Exercise

```
fhr = open('./imdb.csv', 'r', encoding='utf-8', errors='ignore')

# {director:{title:rating}}
# {'Mel Gibson': {'The Passion of the Christ': '7.1', 'Braveheart': '8.4'},
#  'Alfred Hitchcock': {'The Trouble with Harry': '7.2'}}
imdbND = {}
fhr.readline()
for line in fhr:
    ls = line.split(',')
    director = (ls[1]).strip()
    title = (ls[11]).strip()
    rating = (ls[25]).strip()
    if (director in imdbND.keys()):
        titleratings = imdbND[director]
    else:
        titleratings = {}
    titleratings[title] = rating
    imdbND[director] = titleratings
print (imdbND)

fhr.close()
```

Exercise

```
fhr = open('./dmr.csv', 'w', encoding='utf-8', errors='ignore')

fhr.write("Director,Title,Rating\n")

for director, movieratings in imdbND.items():
    for movie, rating in movieratings.items():
        fhr.write(director + "," + movie + "," + rating + "\n")

fhr.close()
```

Nested Dictionaries and List of Tuples

Download and Use ExerciseRatings.py

```python
def minkowksiD(ratings1, ratings2, r):

    distance = 0
    for item in ratings1.keys():
        if item in ratings2.keys():
            x = ratings1[item]
            y = ratings2[item]
            distance += pow(abs(x - y), r)

    return pow(distance,1/r)


UserMovieRatings = {
    'Amy': {'Family Plot':10, 'Rebecca':5, 'Spellbound':9, 'Star Trek':6},
    'Bill': {'Apocalypto':8, 'Braveheart':3, 'Rebecca':10, 'Spellbound':5, 'Star Trek':7},
    'Cathy': {'Spaceballs':7, 'The Ice Storm':4, 'Family Plot':5, 'Rebecca':9, 'Spellbound':1},
    'Dave': {'Braveheart':5, 'Rebecca':7, 'Spellbound':4},
    'Ernie': {'Apocalypto':3, 'Braveheart':8, 'Rebecca':1, 'Star Trek':7},
    'Fiona': {'The Ice Storm':3, 'Family Plot':10, 'Rebecca':6, 'Spellbound':10}}
```

Exercise

Code up the following:

- Set UserX to ‘Amy’
- Set UserXRatings to movie ratings associated with Amy

```
UserX = 'Amy'
UserXRatings = UserMovieRatings [UserX]
```

Exercise

Code up the following:

- Set UserX to ‘Amy’
- Set UserXRatings to Amy’s movie ratings
- Set UserY to ‘Bill’
- Set UserYRatings to Bill’s movie ratings

```
UserX = 'Amy'
UserXRatings = UserMovieRatings [UserX]

UserY = 'Bill'
UserYRatings = UserMovieRatings [UserY]
```

Exercise

Code up the following:

- Set UserX to ‘Amy’
- Set UserXRatings to Amy’s movie ratings
- Set UserY to ‘Bill’
- Set UserYRatings to Bill’s movie ratings
- Find and print the Manhattan Distance between UserX’s movie ratings and UserY’s movie ratings

```
UserX = 'Amy'
UserXRatings = UserMovieRatings [UserX]

UserY = 'Bill'
UserYRatings = UserMovieRatings [UserY]

manXY = minkowksiD(UserXRatings,UserYRatings,1)
print (manXY)
```

Exercise

Code up the following:

- Write a for loop to print each user, and movie ratings associated with each user

```python
for UserY, UserYRatings in UserMovieRatings.items():
    print (UserY)
    print (UserYRatings)
```

Exercise

Code up the following:

- Set UserX to ‘Amy’
- Set UserXRatings to Amy’s movie ratings
- Use a for loop to find and print the Manhattan Distance between UserX and each of the other users

```
UserX = 'Amy'
UserXRatings = UserMovieRatings [UserX]

for UserY, UserYRatings in UserMovieRatings.items():
    manXY = minkowksiD(UserXRatings,UserYRatings,1)
    print (UserX, UserY, manXY)
```

Exercise

Code up the following:

- Set UserX to ‘Amy’
- Set UserXRatings to Amy’s movie ratings
- Use a for loop to create a list of tuples, where each element in the list is of the form (UserY, manXY)
- Print the list of tuples

```python
UserX = 'Amy'
UserXRatings = UserMovieRatings [UserX]

lst = []
for UserY, UserYRatings in UserMovieRatings.items():
    manXY = minkowksiD(UserXRatings,UserYRatings,1)
    tup = (UserY, manXY)
    lst.append(tup)
print (lst)
```

Exercise

Code up the following:

- Set UserX to ‘Amy’
- Set UserXRatings to Amy’s movie ratings
- Use a for loop to create a list of tuples, where each element in the list is of the form (UserY, manXY)
- Print the list of tuples
- Sort the list of tuples in ascending order by Manhattan distance
- Print the sorted list of tuples

```python
from operator import itemgetter

UserX = 'Amy'
UserXRatings = UserMovieRatings [UserX]

lst = []
for UserY, UserYRatings in UserMovieRatings.items():
    manXY = minkowksiD(UserXRatings,UserYRatings,1)
    tup = (UserY, manXY)
    lst.append(tup)

print (lst)
sortedlst = sorted(lst, key=itemgetter(1))
print (sortedlst)
```