Python Class Exercise Set 3

Exercise

Type in below as-is and see what gets printed

```
# variation 1
for i in range (5):
    print ("hello")

# variation 2
for i in range (5):
    print (i)

# variation 3
for i in range (5):
    print (i)
    print ("hello")
```

Exercise

Type in below as-is and see what gets printed

```
strg = "Sundevils"

for i in range(len(strg)):
    print (i, strg[i])
```

Exercise

Type in below as-is and see what gets printed

```
lst = [10, 20, 30]

# method 1
for i in range (len(lst)):
    print (i, lst[i])

# method 2
for l in lst:
    print (l)
```

Exercise

Type in below as-is and see what gets printed

```
mylst1 = [10, 20, 30]
mylst2 = [100, 200, 300]

# method 1
for i in range (len(mylst1)):
    print (i, mylst1[i], mylst2[i])

# method 2
for l1, l2 in zip (mylst1, mylst2):
    print (l1, l2)
```

Exercise

Type in below as-is and see what gets printed

```
mydictionary = {'A':100, 'B':200, 'C':300}

# variation 1
for k in sorted(mydictionary.keys()):
    print (k, mydictionary[k])

# variation 2
for k in reversed(sorted(mydictionary.keys())):
    print (k, mydictionary[k])
```

Exercise

Type in below as-is and see what gets printed

```
sumi = 0

for i in range (5):
    sumi = sumi + i
    print (i, sumi)

print (sumi)
```

Exercise

- Let’s say you had a list:

  P = [10, 20, 30]

- Use a “for” loop with the range function to find the sum of the elements of the list.

- Print the index, list element, and sum in each iteration of the “for” loop

- Print the final sum.

Answer

```
P = [10, 20, 30]

sumP = 0

for i in range (3):
    sumP = sumP + P[i]
    print (i, P[i], sumP)

print (sumP)
```

Exercise

- Let's sat you had two lists:

  P = [1, 2, 3]

  Q = [10, 20, 30]

- Use a **single** "for" loop with the range function to find the sums of the elements of the two lists.

- Print the two lists, and sum of the elements for the two lists, in user-friendly format

Answer

```
P = [1, 2, 3]
Q = [10, 20, 30]

sumP = 0
sumQ = 0

for i in range (3):
    sumP = sumP + P[i]
    sumQ = sumQ + Q[i]

print ("P =", P)
print ("Q =", Q)
print ("sumP =", sumP)
print ("sumQ =", sumQ)
```

Exercise

- Let's sat you had two lists:

  P = [1, 2, 3]

  Q = [10, 20, 30]

- Use a **single** "for" loop with the range function to find the Manhattan distance between the two lists.

- Print the two lists, Manhattan distance between the two lists, in user-friendly format

Answer

```
P = [1, 2, 3]
Q = [10, 20, 30]

manhattan = 0

for i in range (3):
    manhattan = manhattan + math.fabs(P[i] - Q[i])

print ("P =", P)
print ("Q =", Q)
print ("Manhattan Distance =", round(manhattan,2))
```

Exercise

- Let's sat you had two lists:

  P = [1, 2, 3]

  Q = [10, 20, 30]

- Use a **single** "for" loop with the range function to find the Manhattan, Euclidean, and Minkowski (r=3) distances between the two lists.

- Print the two lists, and the Manhattan, Euclidean and Minkowski (r=3) distances between the two lists, in user-friendly format

Answer

```
P = [1, 2, 3]
Q = [10, 20, 30]

manhattan = 0
euclidean = 0
minkowski = 0

for i in range (len(P)):
    manhattan = manhattan + math.fabs(P[i] - Q[i])
    euclidean = euclidean + pow(math.fabs(P[i] - Q[i]), 2)
    minkowski = minkowski + pow(math.fabs(P[i] - Q[i]), 3)

manhattan = pow (manhattan, 1/1)
euclidean = pow (euclidean, 1/2)
minkowski = pow (minkowski, 1/3)

print ("Manhattan Distance =", round(manhattan,2))
print ("Euclidean Distance =", round(euclidean,2))
print ("Minkowski Distance (r=3) =", round(minkowski,2))
```

Exercise

- Let's say we have two lists:
  - P = [1, 2, 3, 4, 5]
  - Q = [10, 20, 30, 40, 50]

- **Pearson Correlation** (computationally efficient form):

$$r = \frac{\sum_{i=1}^{n} p_i q_i - \frac{\sum_{i=1}^{n} p_i \sum_{i=1}^{n} q_i}{n}}{\sqrt{\sum_{i=1}^{n} p_i{}^2 - \frac{\left(\sum_{i=1}^{n} p_i\right)^2}{n}} \sqrt{\sum_{i=1}^{n} q_i{}^2 - \frac{\left(\sum_{i=1}^{n} q_i\right)^2}{n}}}$$

- Use a single for loop and zip function to calculate the Pearson Correlation between the two lists.
  - *Answer: 1.0*

Answer

```
P = [1, 2, 3, 4, 5]
Q = [10, 20, 30, 40, 50]
n = len(P)

# initialize various component sums
sumpq = 0
sump = 0
sumq = 0
sump2 = 0
sumq2 = 0

# calcualte pearson correlation using the computationally efficient form
for p, q in zip(P, Q):
    sumpq += p * q
    sump += p
    sumq += q
    sump2 += pow(p, 2)
    sumq2 += pow(q, 2)

# pearson correlation coefficient
nr = (sumpq - (sump * sumq) / n)
dr = (math.sqrt(sump2 - pow(sump, 2) / n) *
        math.sqrt(sumq2 - pow(sumq, 2) / n))
r = nr/dr

print ("P =", P)
print ("Q =", Q)
print ("Pearson Correlation =", round(r,2))
```

Exercise

- Let's say User X has rated 5 items (A, B, C, D, E):
  - UserXRatingsD = {'A':1, 'B':2, 'C':3, 'D':4, 'E':5}

- Write a for loop to iterate through the ratings dictionary and print the items and ratings, with keys in sorted order.

  Answer:

  A *1*
  B *2*
  C *3*
  D *4*
  E *5*

Answer

```
UserXRatingsD = {'A':1, 'B':2, 'C':3, 'D':4, 'E':5}

for k in sorted(UserXRatingsD.keys()):
    print(k, UserXRatingsD[k])
```

Exercise

- Let's say Users X and Y have rated 5 items (A, B, C, D, E):
  - UserXRatingsD = {'A':1, 'B':2, 'C':3, 'D':4, 'E':5}
  - UserYRatingsD = {'A':10, 'B':20, 'C':30, 'D':40, 'E':50}

- Use a single for loop to iterate through the ratings dictionary and print the items and ratings, with keys in sorted order.

  Answer:

  A 1 10
  B 2 20
  C 3 30
  D 4 40
  E 5 50

Answer

```
UserXRatingsD = {'A':1, 'B':2, 'C':3, 'D':4, 'E':5}
UserYRatingsD = {'A':10, 'B':20, 'C':30, 'D':40, 'E':50}

for k in sorted(UserXRatingsD.keys()):
    print(k, UserXRatingsD[k], UserYRatingsD[k])
```

Exercise

- Let's say Users X and Y have rated 5 items (A, B, C, D, E):
  - UserXRatingsD = {'A':1, 'B':2, 'C':3, 'D':4, 'E':5}
  - UserYRatingsD = {'A':10, 'B':20, 'C':30, 'D':40, 'E':50}

- Use a single for loop to iterate through the ratings and find the Manhattan Distance between the ratings.

  Answer: 135.0

Answer

```
UserXRatingsD = {'A':1, 'B':2, 'C':3, 'D':4, 'E':5}
UserYRatingsD = {'A':10, 'B':20, 'C':30, 'D':40, 'E':50}

md = 0
for k in UserXRatingsD.keys():
    md = md + math.fabs(UserXRatingsD[k] - UserYRatingsD[k])

print ("UserXRatingsD =", UserXRatingsD)
print ("UserYRatingsD =", UserYRatingsD)
print ("Manhattan Distance =", round(md,2))
```

Exercise

- Let's say Users X and Y have rated 5 items (A, B, C, D, E):
  - UserRatingsND = {'X': {'A':10, 'B':20, 'C':30, 'D':40, 'E':50},
    'Y': {'A':100, 'B':200, 'C':300, 'D':400, 'E':500}}

- Assign the User X item-ratings to a dictionary called UserXRatingsD, and the User Y item-ratings to a dictionary called UserYRatingsD.

- Use a single for loop to iterate through UserXRatingsD and print the items and ratings, with keys in sorted order.

  Answer:

A	*10*	*100*
B	*20*	*200*
C	*30*	*300*
D	*40*	*400*
E	*50*	*500*

Answer

```
UserRatingsND = {'X':{'A':10, 'B':20, 'C':30, 'D':40, 'E':50},
                 'Y':{'A':100, 'B':200, 'C':300, 'D':400, 'E':500}}

UserXRatingsD = UserRatingsND['X']
UserYRatingsD = UserRatingsND['Y']

for k in sorted(UserXRatingsD.keys()):
    print(k, UserXRatingsD[k], UserYRatingsD[k])
```