Ryan Zhang

**CT Unit 7: Algorithms for Data Analysis Project**

**Problem Description/Code/Explanation/Output**

**Task One:** Create a sorted employee data file based on the ascending alphabet order of their last name.

**Code:**

*# Bubble sort the list according to last name*

for i in range(0, len(data)):

    for j in range(0, len(data)-1):

        if data[j]['last'] > data[j+1]['last']:

            data[j], data[j+1] = data[j+1], data[j]

        else:

            continue

pretty\_print(data)

*# Output a file*

print("File is also outputted as 'task\_1.txt'")

with open('task\_1.txt', 'w') as outf:

    for i in data:

        outf.write(**f**"{i['last']} {i['first']} {i['position']} {i['state']} {i['age']} {i['salary']} \n")

**Explanation:**

My approach to this task was to use a bubble sort. A bubble sort is where each index of the list is compared to the next index. If the next index was greater than the previous index, then the two indexes would be swapped in the list. This process would need to be repeated for every index in the list therefore there are two for loops. The time complexity is O(n^2).

**Output:**

As seen in the code the output is written into a .txt file because it is too large to display.

**Task Two:** Print the data of top 5 highest salary employees in Iowa.

**Code:**

*# Gather all the salaries in Iowa*

iowa\_lst = []

for i in data:

    if i['state'] == 'IA':

        iowa\_lst.append(i)

    else:

        continue

*# Bubble sort the list according to salaries*

for j in range(0, len(iowa\_lst)):

    for i in range(0, len(iowa\_lst)-1):

        if iowa\_lst[i+1]['salary'] < iowa\_lst[i]['salary']:

            iowa\_lst[i], iowa\_lst[i+1] = iowa\_lst[i+1], iowa\_lst[i]

        else:

            continue

*# Select the top five*

top\_5 = iowa\_lst[-5:]

pretty\_print(top\_5)

**Explanation:**

First, because all my data was stored in a list of dictionaries (each dictionary representing one employee), I needed to gather all the employees into a new list where the only employees are in the state of Iowa. To do this I simply which through each element in the list and appended them to *iowa\_lst* if they were from Iowa. For sorting the salaries, I used a bubble sort. A bubble sort is where each index of the list is compared to the next index. If the next index was greater than the previous index, then the two indexes would be swapped in the list. This process would need to be repeated for every index in the list therefore there are two for loops. The time complexity is O(n^2). Then at the very end I just sliced the list to reveal the *top\_5* highest paid salary.

**Output:**

{'last': 'Wood', 'first': 'Graham', 'position': 'Staff', 'state': 'IA', 'age': 76, 'salary': 94000}

{'last': 'Nichols', 'first': 'Alexa', 'position': 'Staff', 'state': 'IA', 'age': 62, 'salary': 101000}

{'last': 'Marek', 'first': 'Natalie', 'position': 'Staff', 'state': 'IA', 'age': 46, 'salary': 103000}

{'last': 'Johnson', 'first': 'Angela', 'position': 'Staff', 'state': 'IA', 'age': 71, 'salary': 107000}

{'last': 'Biddle', 'first': 'Alejandra', 'position': 'Sr.Staff', 'state': 'IA', 'age': 73, 'salary': 127000}

**Task Three:** Find the data of an employee whose first name is Sebastian, and he is 53 years old. (Who is he?)

**Code:**

*# Search the entire list for Sebastian*

for i in data:

    if i['first'] == 'Sebastian' and i['age'] == 53:

        sebastian = i

    else:

        continue

print(**f**"This is Sebastian: {sebastian}")

**Explanation:**

For this task I simply went through every element of list and searched for a Sebastian who was also 53 years old.

Output:

This is Sebastian: {'last': 'Thrun', 'first': 'Sebastian', 'position': 'Director', 'state': 'CA', 'age': 53, 'salary': 750000}

**Task Four:** Compare the average salaries of Senior Staff and Junior Staff.

**Code:**

*# Gather all the sr and jr salaries into list*

sr\_lst = []

jr\_lst = []

for i in data:

    if i['position'] == 'Sr.Staff':

        sr\_lst.append(i['salary'])

    elif i['position'] == 'Jr.Staff':

        jr\_lst.append(i['salary'])

    else:

        continue

print(**f**"Average of Sr.Staff: {mean(sr\_lst)}")

print(**f**"Average of Jr.Staff: {mean(jr\_lst)}")

**Explanation:**

For this task I went through each element in the data and appended whoever was a Sr. or Jr. Staff’s salaries into their respective lists. This I just averaged the salaries.

**Output:**

Average of Sr.Staff: 127548.3870967742

Average of Jr.Staff: 47555.77299412916