**Assignment**

**Data Structures**

*Announced Date: 4th Nov, 2024 Due Date: 5th Jan, 2024 Total Marks: 10*

**Bubble Sort Exercise:**

Modify bubble\_sort function such that it can sort following list of transactions happening in an electronic store,

*elements = [*

*{ 'name': 'Hamza', 'transaction\_amount': 1000, 'device': 'iphone-10'},*

*{ 'name': 'Hasnain', 'transaction\_amount': 400, 'device': 'google pixel'},*

*{ 'name': 'Palwasha', 'transaction\_amount': 200, 'device': 'vivo'},*

*{ 'name': 'Aamir', 'transaction\_amount': 800, 'device': 'iphone-8'},*

*]*

bubble\_sort function should take key from a transaction record and sort the list as per that key. For example,

*bubble\_sort(elements, key='transaction\_amount')*

This will sort elements by transaction\_amount and your sorted list will look like,

*elements = [*

*{ 'name': ' Palwasha ', 'transaction\_amount': 200, 'device': 'vivo'},*

*{ 'name': ' Hasnain ', 'transaction\_amount': 400, 'device': 'google pixel'},*

*{ 'name': 'Aamir', 'transaction\_amount': 800, 'device': 'iphone-8'},*

*{ 'name': ' Hamza ', 'transaction\_amount': 1000, 'device': 'iphone-10'},*

*]*

But if you call it like this,

*bubble\_sort(elements, key='name')*

output will be,

*elements = [*

*{ 'name': 'aamir', 'transaction\_amount': 800, 'device': 'iphone-8'},*

*{ 'name': ' Hamza ', 'transaction\_amount': 1000, 'device': 'iphone-10'},*

*{ 'name': ' Hasnain ', 'transaction\_amount': 400, 'device': 'google pixel'},*

*{ 'name': ' Palwasha ', 'transaction\_amount': 200, 'device': 'vivo'},*

*]*

**Selection Sort Exercise:**

Implement a Multi-Level Sort of a given list of dictionaries based on a given sorting order. If user wants to sort dictionary based on First Key 'A', Then Key 'B', they shall pass list of keys in the order of preference as a list ['A','B']. Your code should be able to sort list of dictionaries for any number of keys in sorting order list.

Using this multi-level sort, you should be able to sort any list of dictionaries based on sorting order preference

Example: A single dictionary entry contains two keys 'First Name' and 'Last Name'. the list should be sorted first based on 'First Name', then based on 'Last Name', w.r.t. common/same 'First Name' entries.

For this, one shall pass sorting order of preference list [ 'First Name’, 'Last Name' ]

For this, Given the following sequence List:

*[*

*{'First Name': 'Hamza', 'Last Name': 'Farooqui'},*

*{'First Name': 'Junaid', 'Last Name': 'Qamar'},*

*{'First Name': 'Farooq', 'Last Name': 'Wajih'},*

*{'First Name': 'Jade', 'Last Name': 'Canary'},*

*{'First Name': 'Mohammad', 'Last Name': 'Ali'},*

*{'First Name': 'Mohammad', 'Last Name': 'Ahmed'},*

*{'First Name': 'Kiran', 'Last Name': 'Kamla'},*

*{'First Name': 'Armaan', 'Last Name': 'Sheikh'},*

*{'First Name': 'Shaheen', 'Last Name': 'Afridi'},*

*{'First Name': 'Ingrid', 'Last Name': 'Galore'},*

*{'First Name': 'Shahid', 'Last Name': 'Afridi'},*

*{'First Name': 'Armaan', 'Last Name': 'Seth'},*

*{'First Name': 'Ingrid', 'Last Name': 'Maverick'},*

*{'First Name': 'Aahana', 'Last Name': 'Arora'}*

*]*

Your algorithm should generate sorted list:

*[*

*{'First Name': 'Aahana', 'Last Name': 'Arora'},*

*{'First Name': 'Armaan', 'Last Name': 'Seth'},*

*{'First Name': 'Armaan', 'Last Name': 'Sheikh'},*

*{'First Name': 'Farooq', 'Last Name': 'Wajih'},*

*{'First Name': 'Hamza', 'Last Name': 'Farooqui'},*

*{'First Name': 'Ingrid', 'Last Name': 'Galore'},*

*{'First Name': 'Ingrid', 'Last Name': 'Maverick'},*

*{'First Name': 'Jade', 'Last Name': 'Canary'},*

*{'First Name': 'Junaid', 'Last Name': 'Qamar'},*

*{'First Name': 'Kiran', 'Last Name': 'Kamla'},*

*{'First Name': 'Mohammad', 'Last Name': 'Ahmed'},*

*{'First Name': 'Mohammad', 'Last Name': 'Ali'},*

*{'First Name': 'Shaheen', 'Last Name': 'Afridi'},*

*{'First Name': 'Shahid', 'Last Name': 'Afridi'}*

*]*

**Insertion Sort Exercise:**

Compute the running median of a sequence of numbers. That is, given a stream of numbers, print out the median of the list so far on each new element.

Recall that the median of an even-numbered list is the average of the two middle numbers in a *sorted list*.

For example, given the sequence [2, 1, 5, 7, 2, 0, 5], your algorithm should print out:

2

1.5

2

3.5

2

2

2

**Quick Sort Exercise:**

Implement quick sort using Hoare partition scheme. This partition scheme takes the first element of the array as the pivot, meawhile the implementation remains same. You need to write python code to implement it.

**Merge Sort Exercise:**

Modify merge\_sort function such that it can sort following list of athletes as per the time taken by them in the marathon,

*elements = [*

*{ 'name': 'Riley', 'age': 17, 'time\_hours': 1},*

*{ 'name': 'Jimmy', 'age': 12, 'time\_hours': 3},*

*{ 'name': 'Zack', 'age': 21, 'time\_hours': 2.5},*

*{ 'name': 'Mike', 'age': 24, 'time\_hours': 1.5},*

*]*

merge\_sort function should take key from an athlete's marathon log and sort the list as per that key. For example,

*merge\_sort(elements, key='time\_hours', descending=True)*

This will sort elements by time\_hours and your sorted list will look like,

*elements = [*

*{'name': ' Jimmy', 'age': 12, 'time\_hours': 3},*

*{'name': ' Zack', 'age': 21, 'time\_hours': 2.5},*

*{'name': ' Mike', 'age': 24, 'time\_hours': 1.5},*

*{'name': ' Riley', 'age': 17, 'time\_hours': 1},*

*]*

But if you call it like this,

*merge\_sort(elements, key='name')*

output will be,

*elements = [*

*{ 'name': ' Mike', 'age': 24, 'time\_hours': 1.5},*

*{ 'name': ' Jimmy', 'age': 12, 'time\_hours': 3},*

*{ 'name': ' Riley', 'age': 17, 'time\_hours': 1},*

*{ 'name': ' Zack', 'age': 21, 'time\_hours': 2.5},*

*]*

**General Exercise:**

1. nyc\_weather.csv contains new york city weather for first few days in the month of January. Write a program that can answer following,
   1. What was the average temperature in first week of Jan
   2. What was the maximum temperature in first 10 days of Jan
2. nyc\_weather.csv contains new york city weather for first few days in the month of January. Write a program that can answer following,
   1. What was the temperature on Jan 9?
   2. Wh What was the temperature on Jan 4?
3. poem.txt Contains famous poem "Road not taken" by poet Robert Frost. You have to read this file in python and print every word and its count as show below. Think about the best data structure that you can use to solve this problem and figure out why you selected that specific data structure

*'diverged': 2,*

*'in': 3,*

*'I': 8*