Question 1: [5 marks] Implement (write codes) the Edit DIstance problem discussed in class via **Dynamic Programming** approach both **iteratively and recursively**.

- ✓ Implement iterative **code** using **(part i)** simple 2-D matrix **(part ii)** simple Dictionary **(part iii)** Graph via adjacency matrix **(part iv)** Graph via adjacency list.
- ✓ Are there any major implementation differences between (part i) & (part iii) and similarly between (part ii) & (part iv)? Give your justifications.
- ✓ Dig down google to learn how graph structures can be implemented in python.
- ✓ In case you think a specific data-structure can't be used for implementation; state your clear reasoning.
- ✓ Also, comment how much space each data structure is taking for your implemented solutions.
- ✓ Give snapshots of at least 3 different test cases to verify your correct *recursive* implementation.
- ✓ Submit your .py **or** .cpp files as well as pdf file of the screenshots of your test cases.

Question 2: [5 marks] Modify the 0/1 knapsack algorithm discussed in class to display the object/item numbers as well as the bag worth of the filled knapsack at each filling step.

- ✓ Implement **(write code)** using each of the following data structure and give your comparative analysis of pros and cons of each storage structure **(w.r.t to space efficiency as well as time efficiency)** for your algorithm (i) 2-D matrix (ii) Linked List (iii) Tree (iv) Dictionary (v) Stack (vi) Queue.
- ✓ In case you think a specific data-structure can't be used for implementation; state your clear reasoning.
- ✓ Submit your .py **or** .cpp files.

All the Best!