Data Structures and Algorithms Assignment 4

1. **What is the divide and conquer strategy?**

   Divide and conquer strategy is as follows:
   – Divide the problem instance into two or more smaller instances of the same problem,
   – Solve the smaller instances recursively, and assemble the solutions to form a solution of the original instance.
   – The recursion stops when an instance is reached which is too small to divide.

2. **What is binary search and how does it work?**

   Binary search is an efficient algorithm for finding an item from a sorted list of items. It works by repeatedly dividing in half the portion of the list that could contain the item, until you've narrowed down the possible locations to just one.
   Binary Search Approach: Binary Search is a searching algorithm used in a sorted array by repeatedly dividing the search interval in half. The idea of binary search is to use the information that the array is sorted and reduce the time complexity to O(Log n). The basic steps to perform Binary Search are:

   - Begin with an interval covering the whole array.
   - If the value of the search key is less than the item in the middle of the interval, narrow the interval to the lower half.
   - Otherwise, narrow it to the upper half.
   - Repeatedly check until the value is found or the interval is empty.

3. Explain the distinction between a list and a tuple.

SR.NO.	LIST	TUPLE
1	Lists are mutable	Tuples are immutable
2	Implication of iterations is Time-consuming	The implication of iterations is comparatively Faster
3	The list is better for performing operations, such as insertion and deletion.	Tuple data type is appropriate for accessing the elements
4	Lists consume more memory	Tuple consume less memory as compared to the list
5	Lists have several built-in methods	Tuple does not have many built-in methods.
6	The unexpected changes and errors are more likely to occur	In tuple, it is hard to take place.

4. Can you explain how Python manages memory?

The Python memory manager manages chunks of memory called “Blocks”. A collection of blocks of the same size makes up the “Pool”. Pools are created on Arenas, chunks of 256kB memory allocated on heap=64 pools. If the objects get destroyed, the memory manager fills this space with a new object of the same size.
Methods and variables are created in Stack memory. A stack frame is created whenever methods and variables are created. These frames are destroyed automatically whenever methods are returned.
Objects and instance variables are created in Heap memory. As soon as the variables and functions are returned, dead objects will be garbage collected.
It is important to note that the Python memory manager doesn’t necessarily release the memory back to the Operating System, instead memory is returned back to the python interpreter. Python has a small objects allocator that keeps memory allocated for further use. In long-running processes, you may have an incremental reserve of unused memory.

5. What is the difference between pickling and unpickling?

They are inverses of each other.

Pickling, also called serialization, involves converting a Python object into a series of bytes which can be written out to a file.

Unpicking, or de-serialization, does the opposite–it converts a series of bytes into the Python object it represents.

6. What are the different types of search algorithms?

The searching algorithms are used to search or find one or more than one element from a dataset. These type of algorithms are used to find elements from a specific data structures.

Searching may be sequential or not. If the data in the dataset are random, then we need to use sequential searching. Otherwise we can use other different techniques to reduce the complexity.

Linear Search

A linear search or sequential search is a method for finding an element within a list. This type of searching algorithms sequentially checks each element of the list until a match is found or the whole list has been searched.

A linear search runs in at worst linear time and makes at most n comparisons, where n is the length of the list.

If each element is equally likely to be searched, then linear search has an average case of n+1/2 comparisons, but the average case can be affected if the search probabilities for each element vary.

Linear search is rarely practical because other search algorithms and schemes, such as the binary search algorithm and hash tables, allow significantly faster searching for all but short lists.

Binary Search

This type of searching algorithms is used to find the position of a specific value contained in a sorted array. Binary search algorithm works on the principle of divide & conquer and it is considered the best searching algorithms because of its faster speed to search (Provided the data is in sorted form).

A binary search is also known as a half-interval search or logarithmic search.

It starts by searching in the middle of the array and going down the first lower or upper half of the sequence. If the median value is lower than the target value, that means that the search needs to go higher, if not, then it needs to look on the descending portion of the array.

Binary Search Tree, is a node-based binary tree data structure which has the following properties:

The left subtree of a node contains only nodes with keys lesser than the node's key.
The right subtree of a node contains only nodes with keys greater than the node's key.
The left and right subtree each must also be a binary search tree. There must be no duplicate nodes.
A binary search is a quick and efficient method of finding a specific target value from a set of ordered items. By starting in the middle of the sorted list, it can effectively cut the search space in half by determining whether to ascend or descend the list based on the median value compared to the target value.

For example, with a target value of 8 and a search space of 1 through 11:

The median/middle value is found and the pointer is set there, which in this case is 6.
The target of 8 is compared to 6. Since 6 is smaller than 8, the target must be in the higher half.
The pointer is moved to the next value (7) and compared to the target. It is smaller, therefore the pointer moves to the next higher value.
The pointer is now on 8. Comparing this to the target, it is an exact match, therefore the target has been found.
Using binary search, the target only had to be compared to 3 values. Compared to doing a linear search, it would have started from the very first value and moved up, needing to compare the target to eight values.

A binary search is possible only with an ordered set of data, if the data is randomly arranged, then a linear search would yield results.