CS22203 Lab Assignment no. 3

Ex. For each of the following algorithms, the total no. of element comparisons gives an estimate of the time taken by the algorithm. Modify the algorithms and execute them so that you can: i) count the no. of element comparisons (e), ii) find the actual time taken (in ms) by the program (t). Fill up the respective tables. [You may use time functions in Python for finding the actual time taken]. Plot a graph for each table [you may use line chart or bar chart].

Hint: Fill the list A with numbers automatically using a Python function.

1) Linear Search:

Searching for an element, x is an unsorted list, A of n elements

```
LSearch(A,x,n)
// Returns the position where x is found, otherwise -1.
Begin
for i =1 to n
do
        if x == A[i] then
                return i
        endif
done
return -1
end
```

n	10	100	1000	10000
e (when x is the 1st element)				
e (when x is the last element)				
e (when x is not in the list)				

n	10	100	1000	10000
e (when x is the 1st element)				
e (when x is the last element)				
e (when x is not in the list)				

2) Search for an element, x in an unordered list of n elements using D-and-C

```
Search(x,L)
{
```

```
if |L|==1 // L={e1}, contains only 1 element
        if x== e1, return 1;
        else return -1;
split L into  L1 and L2;
i = Search(x,L1);
if (i !=-1)
        return i;
else {
        j = Search(x,L2);
        if (j !=-1)  return j;
                else return -1;
      }
}
```

n	10	100	1000	10000
e (when x is the 1st element)				
e (when x is the last element)				
e (when x is not in the list)				

n	10	100	1000	10000
t (when x is the 1st element)				
t (when x is the last element)				
t (when x is not in the list)				

3) Search for an element, x in a sorted list of n elements using D-and-C

```
BinSearch(x,L)
{
if |L|==0 return -1;
split L into 3 parts: L1 , middle element and L2;
if x < middle element
        BinSearch(x,L1);
else
        if x > middle element
                BinSearch(x,L2);
        else
                // x is themiddle element
                return mid (i.e., position of middle element);

        endif
```

endif
}

n	10	100	1000	10000
e (when x is the 1st element)				
e (when x is the last element)				
e (when x is not in the list)				

n	10	100	1000	10000
t (when x is the 1st element)				
t (when x is the last element)				
t (when x is not in the list)				
