```
# File: ComparingLinearBinarySearch.py
# Student: Jennifer Truong
# UT EID: Jat5244
# Course Name: CS303E
#
# Date Created: 4/13/2021
# Date Last Modified:  4/16/2021
# Description of Program: Compare the performance of linear and binary search.
# "Comparing" means seeing on average how many probes (comparisons) are made
# as you search a list.

import random, math

# a list of values between 0 and 999
lst = list(range(0, 999))

# Shuffles the values of 0 to 999 randomly
random.shuffle(lst)

# Randomly chooses an integer between 0 and 999
key = random.randint(0, 999 )


def linearSearch( lst, key ):
    """ Search for key in unsorted list lst.  Note that
        the index + 1 is also the number of probes. """
    for i in range( len(lst) ):
        if key == lst[i]:
            return i
    return -1

def binarySearch( lst, key ):
    """ Search for key in sorted list lst. Return
        a pair containing the index (or -low - 1)
        and a count of number of probes. """
    count = 0
    low = 0
    high = len(lst) - 1
    while (high >= low):
        count += 1
        mid = (low + high) // 2
        if key < lst[mid]:
            high = mid - 1
        elif key == lst[mid]:
            return count
        else:
            low = mid + 1
    # Search failed
    return count


def main():
    linearSearchesProbes = []
    averageLinear = []
    print("Linear search:")

    for n in [ 10, 100, 1000, 10000, 100000]:                  # Iterating through
the list
        for searches in range (n):                             # Searches or
```

```
implement the linear search function n times
            key = random.randint(0, 999)                          # Changes the key
after every search
            linearSearchesProbes.append( linearSearch( lst, key ) + 1)
        averageLinear.append( sum( linearSearchesProbes ) / n )
        print("  Tests: {0:<8} Average probes: {1}".format( n, ''.join(map(str,
averageLinear)))) # Prints/ formats the results
        linearSearchesProbes.clear()
        averageLinear.clear()


    test = 0
    binarySearchesProbes = []
    # Looping through 1000 tests and averaging it
    while test < 1000:
        key = random.randint(0, 999 )
        binarySearchesProbes.append( binarySearch( lst, key ) )
        test += 1
    averageBinary = sum( binarySearchesProbes ) / 1000
    differences = ( math.log ( 1000 , 2 ) ) - averageBinary         # Difference
between the log base 2 of 1000 & my average
    print("Binary search:")
    print("  Average number of probes: " + str(averageBinary) )
    print("  Differs from log2(1000) by: " + str(differences) )


main()
```