CMPSC 412 – Lab-1 (25 points)

Due date: 1 week

Name: Wilson Quilli

Goal: To compare the operations of Lists and Dictionaries by calculating and comparing the CPU time taken to perform these operations.

Lab Exercises:

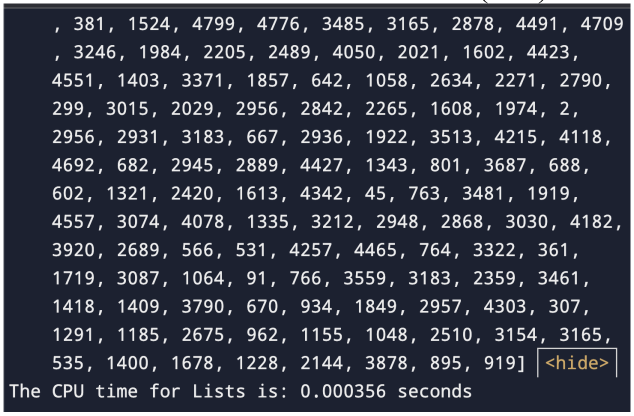
Design a program to generate a specified quantity of integers (minimum 2000 integers). Use rand function to generate integers between 1 and 5000. Use the same dataset for both the data structure. Note that these random numbers can contain duplicates. Store this collection of integers in a) Lists and b) Dictionaries. Perform the following for the List and the Dictionary:

1. Print all the elements. Perform this operation 3 times for each data structure. Measure empirically the time (CPU time) it takes to print all the elements each time. Tabulate your results – Table-1.

#sample of results – screenshots and Table-1 as shown below

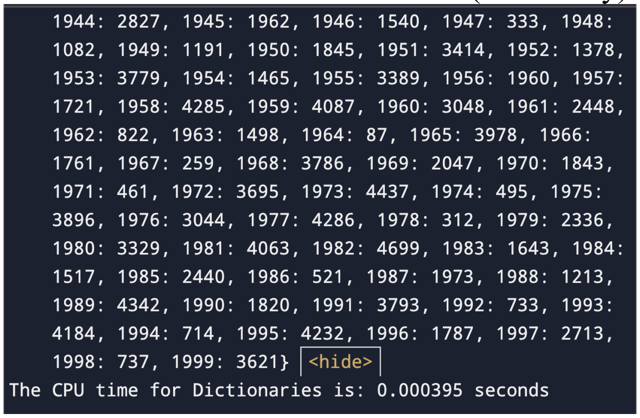
Print operation	List	Dictionary
Trail-1	0.000356 Seconds	0.000395 Seconds
Trail-2	0.000229 Seconds	0.000397 Seconds
Trail-3	0.000189 Seconds	0.000433 Seconds

Screenshot 1: Print all the Elements (List)



```
    , 381, 1524, 4799, 4776, 3485, 3165, 2878, 4491, 4709
    , 3246, 1984, 2205, 2489, 4050, 2021, 1602, 4423,
    4551, 1403, 3371, 1857, 642, 1058, 2634, 2271, 2790,
    299, 3015, 2029, 2956, 2842, 2265, 1608, 1974, 2,
    2956, 2931, 3183, 667, 2936, 1922, 3513, 4215, 4118,
    4692, 682, 2945, 2889, 4427, 1343, 801, 3687, 688,
    602, 1321, 2420, 1613, 4342, 45, 763, 3481, 1919,
    4557, 3074, 4078, 1335, 3212, 2948, 2868, 3030, 4182,
    3920, 2689, 566, 531, 4257, 4465, 764, 3322, 361,
    1719, 3087, 1064, 91, 766, 3559, 3183, 2359, 3461,
    1418, 1409, 3790, 670, 934, 1849, 2957, 4303, 307,
    1291, 1185, 2675, 962, 1155, 1048, 2510, 3154, 3165,
    535, 1400, 1678, 1228, 2144, 3878, 895, 919] <hide>
The CPU time for Lists is: 0.000356 seconds
```

Screenshot 2: Print all the Elements (Dictionary)



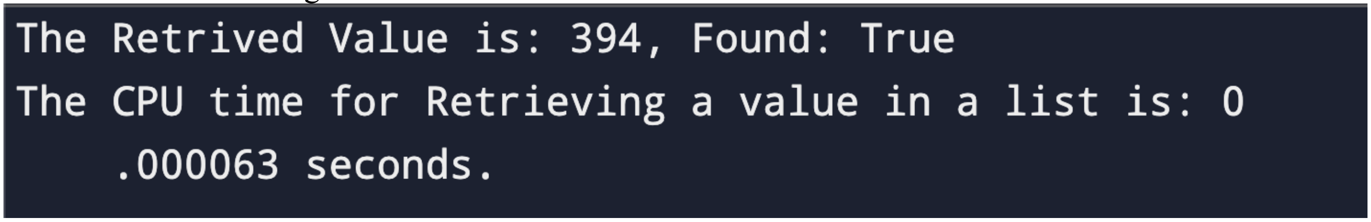
```
    1944: 2827, 1945: 1962, 1946: 1540, 1947: 333, 1948:
    1082, 1949: 1191, 1950: 1845, 1951: 3414, 1952: 1378,
    1953: 3779, 1954: 1465, 1955: 3389, 1956: 1960, 1957:
    1721, 1958: 4285, 1959: 4087, 1960: 3048, 1961: 2448,
    1962: 822, 1963: 1498, 1964: 87, 1965: 3978, 1966:
    1761, 1967: 259, 1968: 3786, 1969: 2047, 1970: 1843,
    1971: 461, 1972: 3695, 1973: 4437, 1974: 495, 1975:
    3896, 1976: 3044, 1977: 4286, 1978: 312, 1979: 2336,
    1980: 3329, 1981: 4063, 1982: 4699, 1983: 1643, 1984:
    1517, 1985: 2440, 1986: 521, 1987: 1973, 1988: 1213,
    1989: 4342, 1990: 1820, 1991: 3793, 1992: 733, 1993:
    4184, 1994: 714, 1995: 4232, 1996: 1787, 1997: 2713,
    1998: 737, 1999: 3621} <hide>
The CPU time for Dictionaries is: 0.000395 seconds
```

2. A series of retrievals (i.e., find) of random values (use rand function) in the collection and measure empirically the time it takes to do the retrievals. Note: use the same value generated from rand function in list and dictionary and find that in the dataset. You should run this program 3 times for each data structure. Tabulate your results – Table-2.

#sample of results – screenshots and Table-2 as shown below

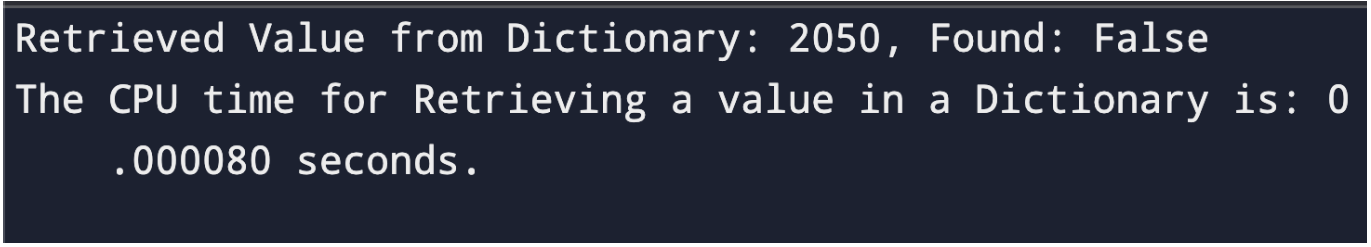
Find operation	List	Dictionary
Trail-1	0.000063 Seconds	0.000114 Seconds
Trail-2	0.000060 Seconds	0.000074 Seconds
Trail-3	0.000053 Seconds	0.000080 Seconds

Screenshot 1: Retrieving a value in a List



```
The Retrived Value is: 394, Found: True
The CPU time for Retrieving a value in a list is: 0
    .000063 seconds.
```

Screenshot 2: Retrieving a value in a Dictionary



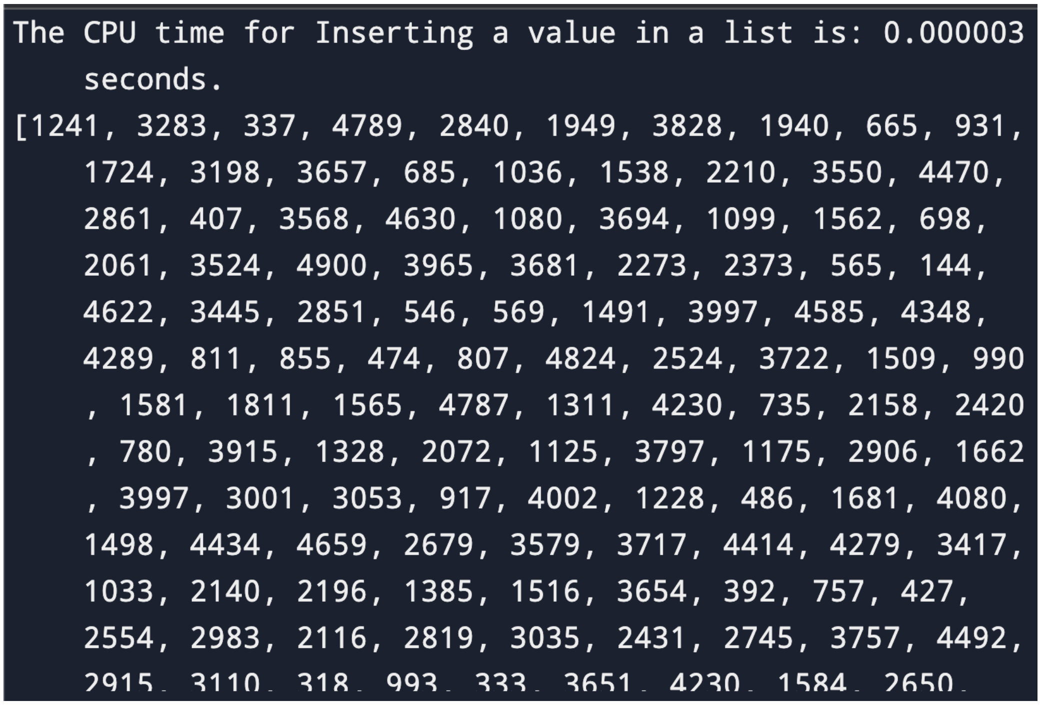
```
Retrieved Value from Dictionary: 2050, Found: False
The CPU time for Retrieving a value in a Dictionary is: 0
    .000080 seconds.
```

3. Generate a random number, perform insertion operation, and print all the elements. Use the same random number for both the data structure. You should run this program 3 times for each data structure. Tabulate your results – Table-3.

#sample of results – screenshots and Table-3 as shown below

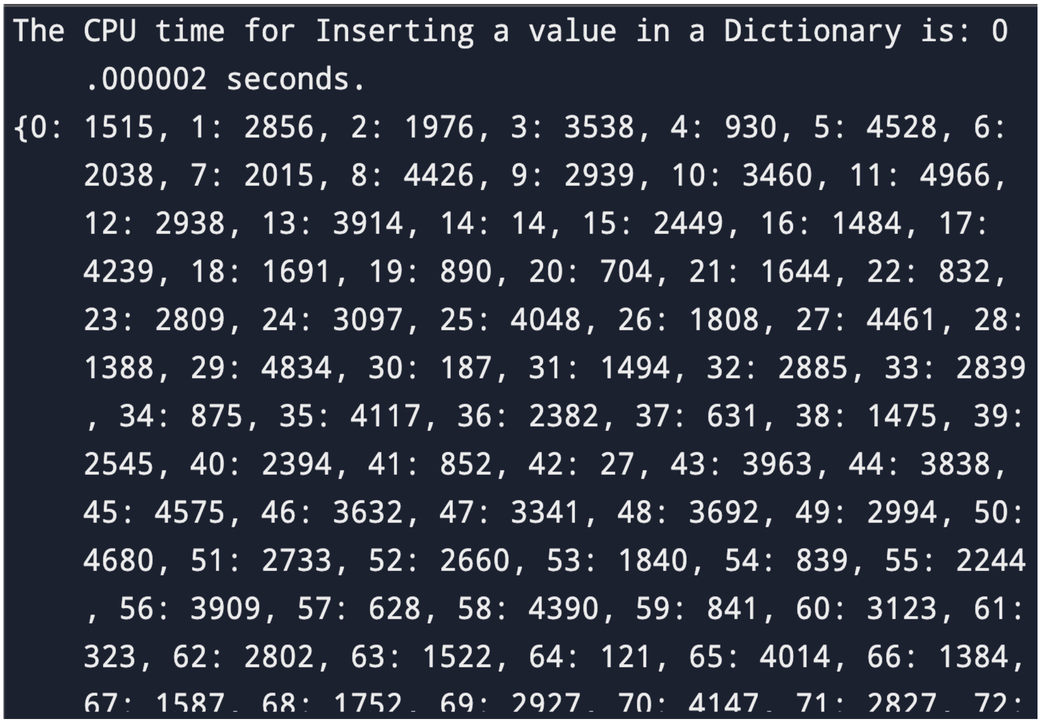
Insert operation	List	Dictionary
Trail-1	0.000003 Seconds	0.000002 Seconds
Trail-2	0.000005 Seconds	0.000002 Seconds
Trail-3	0.000003 Seconds	0.000002 Seconds

Screenshot 1: Inserting a value in a List



```
The CPU time for Inserting a value in a list is: 0.000003
    seconds.
[1241, 3283, 337, 4789, 2840, 1949, 3828, 1940, 665, 931,
    1724, 3198, 3657, 685, 1036, 1538, 2210, 3550, 4470,
    2861, 407, 3568, 4630, 1080, 3694, 1099, 1562, 698,
    2061, 3524, 4900, 3965, 3681, 2273, 2373, 565, 144,
    4622, 3445, 2851, 546, 569, 1491, 3997, 4585, 4348,
    4289, 811, 855, 474, 807, 4824, 2524, 3722, 1509, 990
    , 1581, 1811, 1565, 4787, 1311, 4230, 735, 2158, 2420
    , 780, 3915, 1328, 2072, 1125, 3797, 1175, 2906, 1662
    , 3997, 3001, 3053, 917, 4002, 1228, 486, 1681, 4080,
    1498, 4434, 4659, 2679, 3579, 3717, 4414, 4279, 3417,
    1033, 2140, 2196, 1385, 1516, 3654, 392, 757, 427,
    2554, 2983, 2116, 2819, 3035, 2431, 2745, 3757, 4492,
    2915, 3110, 318, 993, 333, 3651, 4230, 1584, 2650,
```

Screenshot 2: Inserting a value in a Dictionary



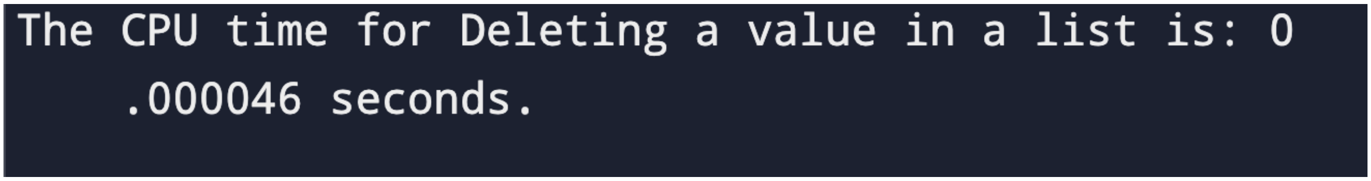
```
The CPU time for Inserting a value in a Dictionary is: 0
    .000002 seconds.
{0: 1515, 1: 2856, 2: 1976, 3: 3538, 4: 930, 5: 4528, 6:
    2038, 7: 2015, 8: 4426, 9: 2939, 10: 3460, 11: 4966,
    12: 2938, 13: 3914, 14: 14, 15: 2449, 16: 1484, 17:
    4239, 18: 1691, 19: 890, 20: 704, 21: 1644, 22: 832,
    23: 2809, 24: 3097, 25: 4048, 26: 1808, 27: 4461, 28:
    1388, 29: 4834, 30: 187, 31: 1494, 32: 2885, 33: 2839
    , 34: 875, 35: 4117, 36: 2382, 37: 631, 38: 1475, 39:
    2545, 40: 2394, 41: 852, 42: 27, 43: 3963, 44: 3838,
    45: 4575, 46: 3632, 47: 3341, 48: 3692, 49: 2994, 50:
    4680, 51: 2733, 52: 2660, 53: 1840, 54: 839, 55: 2244
    , 56: 3909, 57: 628, 58: 4390, 59: 841, 60: 3123, 61:
    323, 62: 2802, 63: 1522, 64: 121, 65: 4014, 66: 1384,
    67: 1587, 68: 1752, 69: 2927, 70: 4147, 71: 2827, 72:
```

4. Generate a random number and perform deletion operation on the list and dictionary data structures. Use the same random number for both the data structure. You should run this program 3 times for each data structure. Tabulate your results – Table-4.

#sample of results – screenshots and Table-4 as shown below

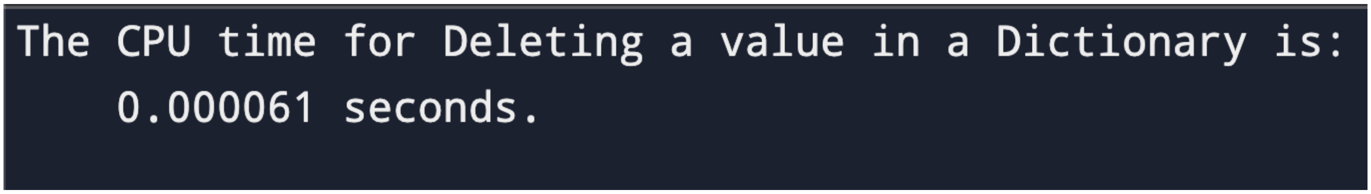
Delete operation	List	Dictionary
Trail-1	0.000045 Seconds	0.000110 Seconds
Trail-2	0.000046 Seconds	0.000061 Seconds
Trail-3	0.000046 Seconds	0.000061 Seconds

Screenshot 1: Deleting a value in a List



```
The CPU time for Deleting a value in a list is: 0
    .000046 seconds.
```

Screenshot 2: Deleting a value in a Dictionary



```
The CPU time for Deleting a value in a Dictionary is:
    0.000061 seconds.
```

Conclusion:

In this lab exercise, I performed four different operations, printing, retrieval, insertion, and deletion, on two data structures, list and dictionary, to compare their efficiency. First, I generated 2000 random values and stored them inside both a list and dictionary, then print out the values, and after performing three trials each, the list is faster. Second, I used the values to then retrieve a random value from each data structure, and as a result the list is faster. Third, after generating 2000 values, I generated another random value and inserted into both the list and dictionary, as a result the dictionary was faster. Finally, after generating 2000 random values and storing in both data structures, I deleted a random value from both, and as a result the list is faster. Overall, the list is more efficient and faster for the 2000 random value dataset in most operations, while the dictionary was only better during insertion. This lab helped me understand how different data structures impact performance depending on the type of operation and supported the idea of choosing the right data structure for each specific tasks.

Sources

- GeeksforGeeks. (2025, July 23). *Python time module*. https://www.geeksforgeeks.org/python/python-time-module/

- *W3schools.com*. W3Schools Online Web Tutorials. (n.d.). https://www.w3schools.com/python/module_random.asp