Wilson Quilli

Professor Elangovan

CMPSC 412: Data Structures Lab

November 17th, 2025

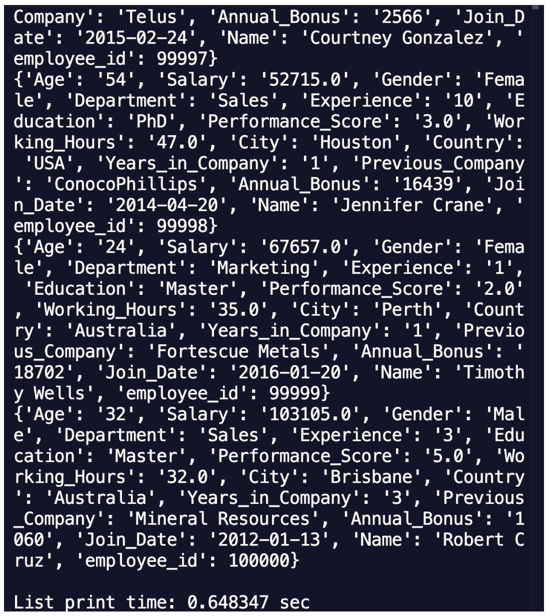
Lab 8 Report: Data Structure Comparison

In this lab, I performed 4 exercises all revolving around four different data structures. The data structures being a list, dictionary, linked list, and a binary search tree. The first exercise being to print out all values from a dataset and comparing how long it takes for each data structure. In this exercise, the linked list appeared to be the fastest of the four because the linked list only moves from one node to the next without extra lookups or indexing. The other data structures have additional processing or pointer checks, so they take slightly more time even though they hold the same data. For the second exercise, I had to retrieve 5 different values on all data structures and compare their times again. In this exercise, the dictionary is the fastest because it can jump straight to the value using the key, so it doesn't have to search through anything. The other data structures move through nodes or elements one by one, which makes them take more time to find the same item. For the third exercise, I had to perform insertion operations in all four data structures. As a result, the linked list is the fastest because adding a new item only needs changing one pointer at the end of the list. The other data structures have to do extra work like resizing, hashing, or finding the right place in the tree, so they take more time. For the final exercise, I had to delete an element from all four data structures and compare the time it took to delete that element. As a result, the dictionary is the fastest because it can remove an item right away using the key without searching through anything. The other data structures must move through the list, tree, or nodes to find the item first, which makes them take longer. From these exercises, I understand from these exercises that every data structure has different strengths depending on the operation being done. Some structures are faster at finding items, others are faster at inserting, and some handle deleting better than others. Overall, the dictionary works the best because it was the fastest in both retrieval and deletion operations. It can look up and remove items instantly using keys, while the other data structures need to search or move through nodes first.

1. Print all the elements.

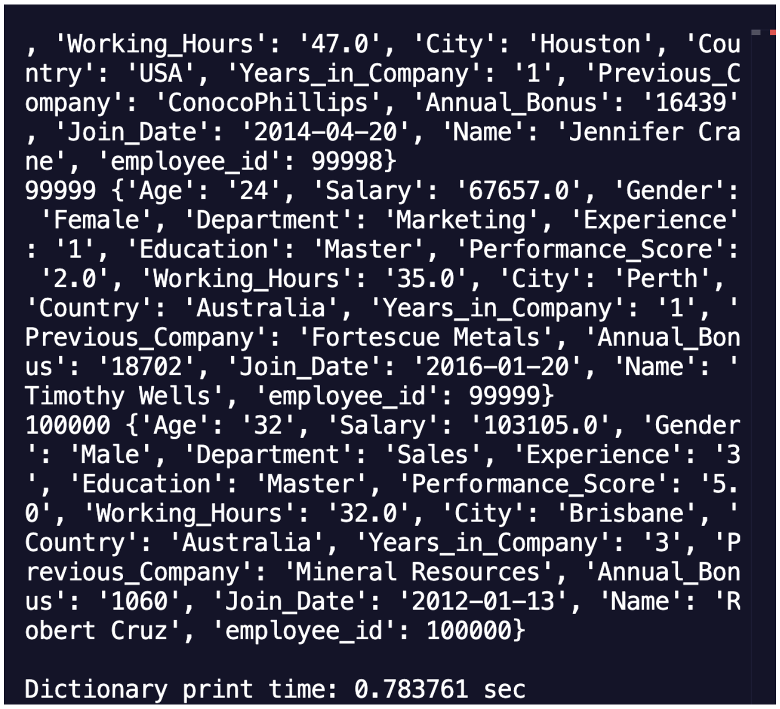
List	Dictionaries	Binary Search Tree	Linked List
0.658347 sec.	0.783761 sec.	0.670789 sec.	0.624194 sec.

Screenshot 1: List - Print



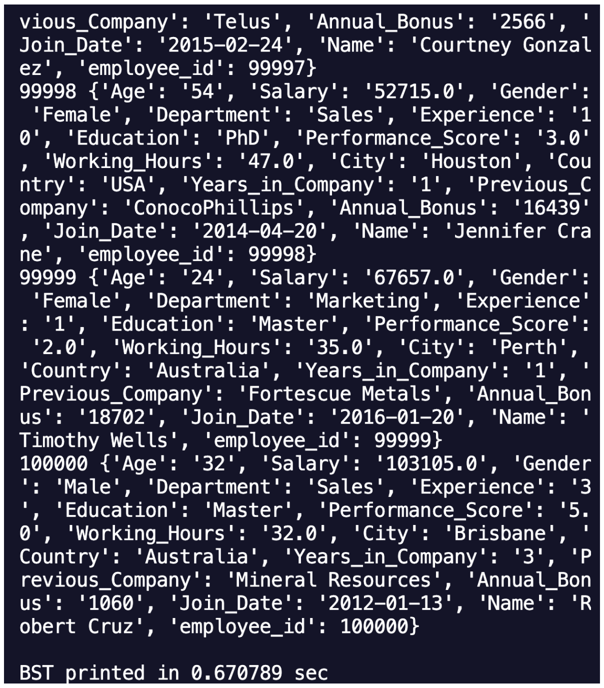
```
Company': 'Telus', 'Annual_Bonus': '2566', 'Join_D
ate': '2015-02-24', 'Name': 'Courtney Gonzalez', '
employee_id': 99997}
{'Age': '54', 'Salary': '52715.0', 'Gender': 'Fema
le', 'Department': 'Sales', 'Experience': '10', 'E
ducation': 'PhD', 'Performance_Score': '3.0', 'Wor
king_Hours': '47.0', 'City': 'Houston', 'Country':
 'USA', 'Years_in_Company': '1', 'Previous_Company
': 'ConocoPhillips', 'Annual_Bonus': '16439', 'Joi
n_Date': '2014-04-20', 'Name': 'Jennifer Crane', '
employee_id': 99998}
{'Age': '24', 'Salary': '67657.0', 'Gender': 'Fema
le', 'Department': 'Marketing', 'Experience': '1',
 'Education': 'Master', 'Performance_Score': '2.0'
, 'Working_Hours': '35.0', 'City': 'Perth', 'Count
ry': 'Australia', 'Years_in_Company': '1', 'Previo
us_Company': 'Fortescue Metals', 'Annual_Bonus': '
18702', 'Join_Date': '2016-01-20', 'Name': 'Timoth
y Wells', 'employee_id': 99999}
{'Age': '32', 'Salary': '103105.0', 'Gender': 'Mal
e', 'Department': 'Sales', 'Experience': '3', 'Edu
cation': 'Master', 'Performance_Score': '5.0', 'Wo
rking_Hours': '32.0', 'City': 'Brisbane', 'Country
': 'Australia', 'Years_in_Company': '3', 'Previous
_Company': 'Mineral Resources', 'Annual_Bonus': '1
060', 'Join_Date': '2012-01-13', 'Name': 'Robert C
ruz', 'employee_id': 100000}

List print time: 0.648347 sec
```

Screenshot 2: Dictionary - Print



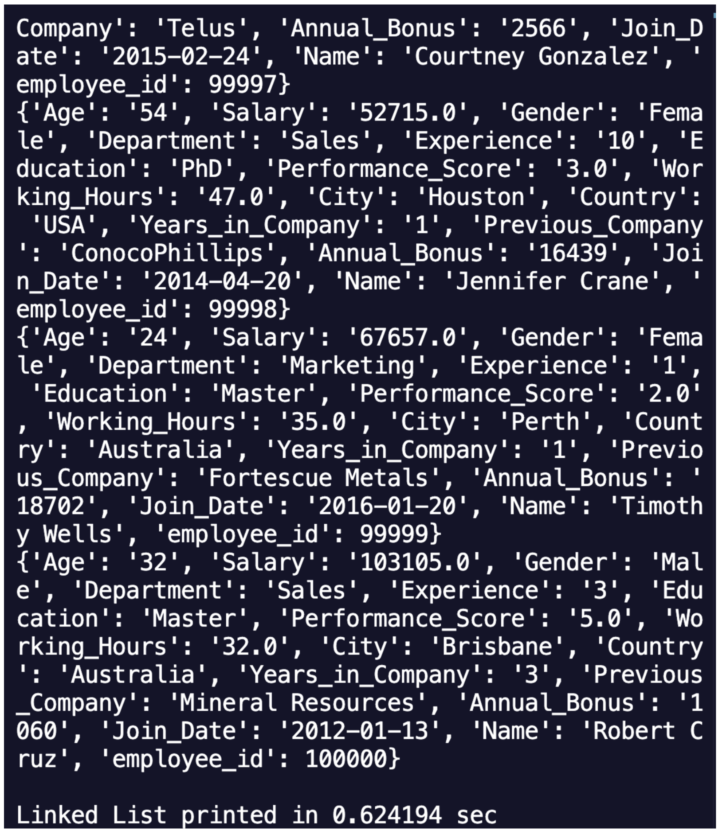
```
, 'Working_Hours': '47.0', 'City': 'Houston', 'Cou
ntry': 'USA', 'Years_in_Company': '1', 'Previous_C
ompany': 'ConocoPhillips', 'Annual_Bonus': '16439'
, 'Join_Date': '2014-04-20', 'Name': 'Jennifer Cra
ne', 'employee_id': 99998}
99999 {'Age': '24', 'Salary': '67657.0', 'Gender':
 'Female', 'Department': 'Marketing', 'Experience'
: '1', 'Education': 'Master', 'Performance_Score':
 '2.0', 'Working_Hours': '35.0', 'City': 'Perth',
'Country': 'Australia', 'Years_in_Company': '1', '
Previous_Company': 'Fortescue Metals', 'Annual_Bon
us': '18702', 'Join_Date': '2016-01-20', 'Name': '
Timothy Wells', 'employee_id': 99999}
100000 {'Age': '32', 'Salary': '103105.0', 'Gender
': 'Male', 'Department': 'Sales', 'Experience': '3
', 'Education': 'Master', 'Performance_Score': '5.
0', 'Working_Hours': '32.0', 'City': 'Brisbane', '
Country': 'Australia', 'Years_in_Company': '3', 'P
revious_Company': 'Mineral Resources', 'Annual_Bon
us': '1060', 'Join_Date': '2012-01-13', 'Name': 'R
obert Cruz', 'employee_id': 100000}

Dictionary print time: 0.783761 sec
```

Screenshot 3: Binary Search Tree - Print



```
vious_Company': 'Telus', 'Annual_Bonus': '2566', '
Join_Date': '2015-02-24', 'Name': 'Courtney Gonzal
ez', 'employee_id': 99997}
99998 {'Age': '54', 'Salary': '52715.0', 'Gender':
 'Female', 'Department': 'Sales', 'Experience': '1
0', 'Education': 'PhD', 'Performance_Score': '3.0'
, 'Working_Hours': '47.0', 'City': 'Houston', 'Cou
ntry': 'USA', 'Years_in_Company': '1', 'Previous_C
ompany': 'ConocoPhillips', 'Annual_Bonus': '16439'
, 'Join_Date': '2014-04-20', 'Name': 'Jennifer Cra
ne', 'employee_id': 99998}
99999 {'Age': '24', 'Salary': '67657.0', 'Gender':
 'Female', 'Department': 'Marketing', 'Experience'
: '1', 'Education': 'Master', 'Performance_Score':
 '2.0', 'Working_Hours': '35.0', 'City': 'Perth',
'Country': 'Australia', 'Years_in_Company': '1', '
Previous_Company': 'Fortescue Metals', 'Annual_Bon
us': '18702', 'Join_Date': '2016-01-20', 'Name': '
Timothy Wells', 'employee_id': 99999}
100000 {'Age': '32', 'Salary': '103105.0', 'Gender
': 'Male', 'Department': 'Sales', 'Experience': '3
', 'Education': 'Master', 'Performance_Score': '5.
0', 'Working_Hours': '32.0', 'City': 'Brisbane', '
Country': 'Australia', 'Years_in_Company': '3', 'P
revious_Company': 'Mineral Resources', 'Annual_Bon
us': '1060', 'Join_Date': '2012-01-13', 'Name': 'R
obert Cruz', 'employee_id': 100000}

BST printed in 0.670789 sec
```

Screenshot 4: Linked List - Print

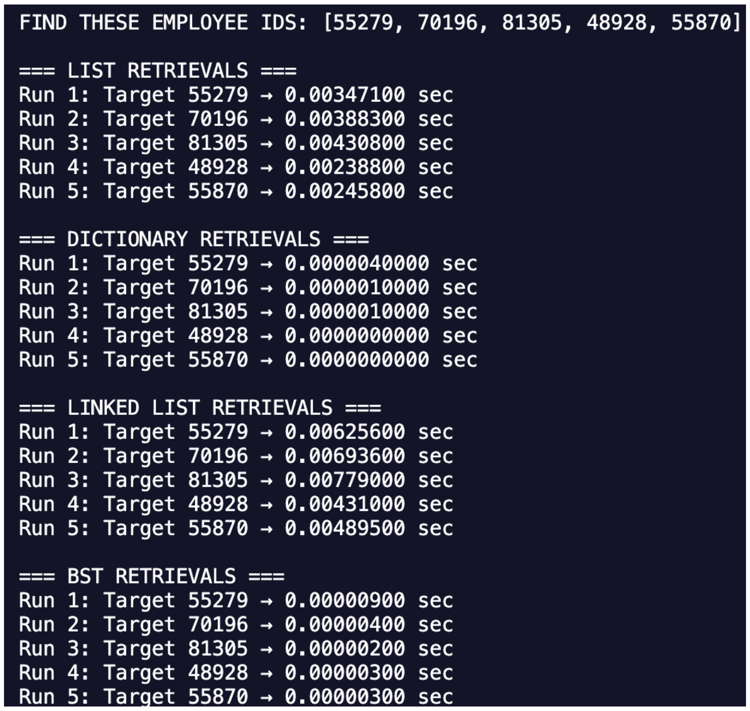


```
Company': 'Telus', 'Annual_Bonus': '2566', 'Join_D
ate': '2015-02-24', 'Name': 'Courtney Gonzalez', '
employee_id': 99997}
{'Age': '54', 'Salary': '52715.0', 'Gender': 'Fema
le', 'Department': 'Sales', 'Experience': '10', 'E
ducation': 'PhD', 'Performance_Score': '3.0', 'Wor
king_Hours': '47.0', 'City': 'Houston', 'Country':
 'USA', 'Years_in_Company': '1', 'Previous_Company
': 'ConocoPhillips', 'Annual_Bonus': '16439', 'Joi
n_Date': '2014-04-20', 'Name': 'Jennifer Crane', '
employee_id': 99998}
{'Age': '24', 'Salary': '67657.0', 'Gender': 'Fema
le', 'Department': 'Marketing', 'Experience': '1',
 'Education': 'Master', 'Performance_Score': '2.0'
, 'Working_Hours': '35.0', 'City': 'Perth', 'Count
ry': 'Australia', 'Years_in_Company': '1', 'Previo
us_Company': 'Fortescue Metals', 'Annual_Bonus': '
18702', 'Join_Date': '2016-01-20', 'Name': 'Timoth
y Wells', 'employee_id': 99999}
{'Age': '32', 'Salary': '103105.0', 'Gender': 'Mal
e', 'Department': 'Sales', 'Experience': '3', 'Edu
cation': 'Master', 'Performance_Score': '5.0', 'Wo
rking_Hours': '32.0', 'City': 'Brisbane', 'Country
': 'Australia', 'Years_in_Company': '3', 'Previous
_Company': 'Mineral Resources', 'Annual_Bonus': '1
060', 'Join_Date': '2012-01-13', 'Name': 'Robert C
ruz', 'employee_id': 100000}

Linked List printed in 0.624194 sec
```

2. Series of Retrievals.

List	Dictionaries	Binary Search Tree	Linked List
0.00347100 sec.	0.000004000 sec.	0.00000900 sec.	0.00625600 sec.
0.00388300 sec.	0.000001000 sec.	0.00000400 sec.	0.00693600 sec.
0.00430800 sec.	0.000001000 sec	0.00000200 sec.	0.00779000 sec.
0.00238800 sec.	0.000001000 sec	0.00000300 sec.	0.00431000 sec.
0.00245800 sec.	0.000003000 sec	0.00000300 sec.	0.00489500 sec.

Screenshot 5: Retrievals

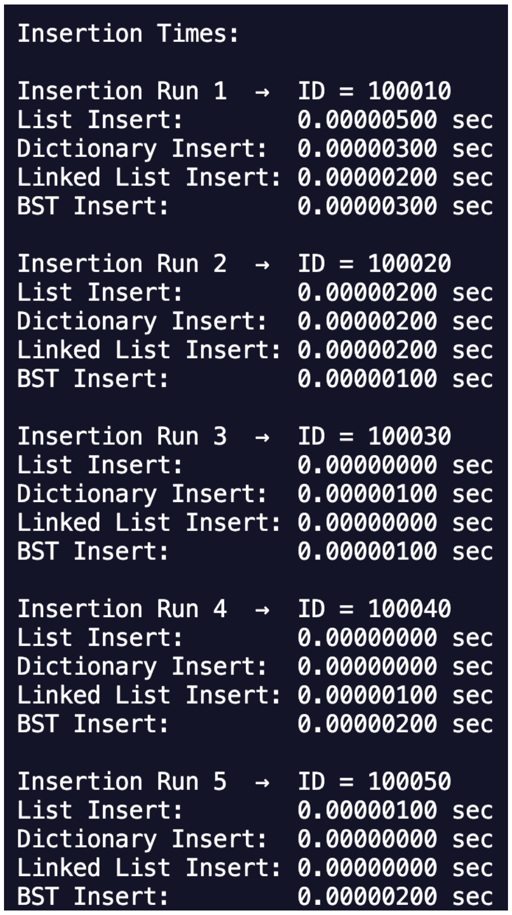


```
FIND THESE EMPLOYEE IDS: [55279, 70196, 81305, 48928, 55870]

=== LIST RETRIEVALS ===
Run 1: Target 55279 → 0.00347100 sec
Run 2: Target 70196 → 0.00388300 sec
Run 3: Target 81305 → 0.00430800 sec
Run 4: Target 48928 → 0.00238800 sec
Run 5: Target 55870 → 0.00245800 sec

=== DICTIONARY RETRIEVALS ===
Run 1: Target 55279 → 0.0000040000 sec
Run 2: Target 70196 → 0.0000010000 sec
Run 3: Target 81305 → 0.0000010000 sec
Run 4: Target 48928 → 0.0000000000 sec
Run 5: Target 55870 → 0.0000000000 sec

=== LINKED LIST RETRIEVALS ===
Run 1: Target 55279 → 0.00625600 sec
Run 2: Target 70196 → 0.00693600 sec
Run 3: Target 81305 → 0.00779000 sec
Run 4: Target 48928 → 0.00431000 sec
Run 5: Target 55870 → 0.00489500 sec

=== BST RETRIEVALS ===
Run 1: Target 55279 → 0.00000900 sec
Run 2: Target 70196 → 0.00000400 sec
Run 3: Target 81305 → 0.00000200 sec
Run 4: Target 48928 → 0.00000300 sec
Run 5: Target 55870 → 0.00000300 sec
```

3. Insertions.

List	Dictionaries	Binary Search Tree	Linked List
0.00000500 sec.	0.00000300 sec.	0.00000300 sec.	0.00000200 sec.
0.00000200 sec.	0.00000200 sec.	0.00000100 sec.	0.00000200 sec.
0.00000300 sec.	0.00000100 sec.	0.00000100 sec.	0.00000200 sec.
0.00000500 sec.	0.00000200 sec.	0.00000200 sec.	0.00000100 sec.
0.00000200 sec.	0.00000100 sec.	0.00000200 sec.	0.00000100 sec.

Screenshot 6: Insertions

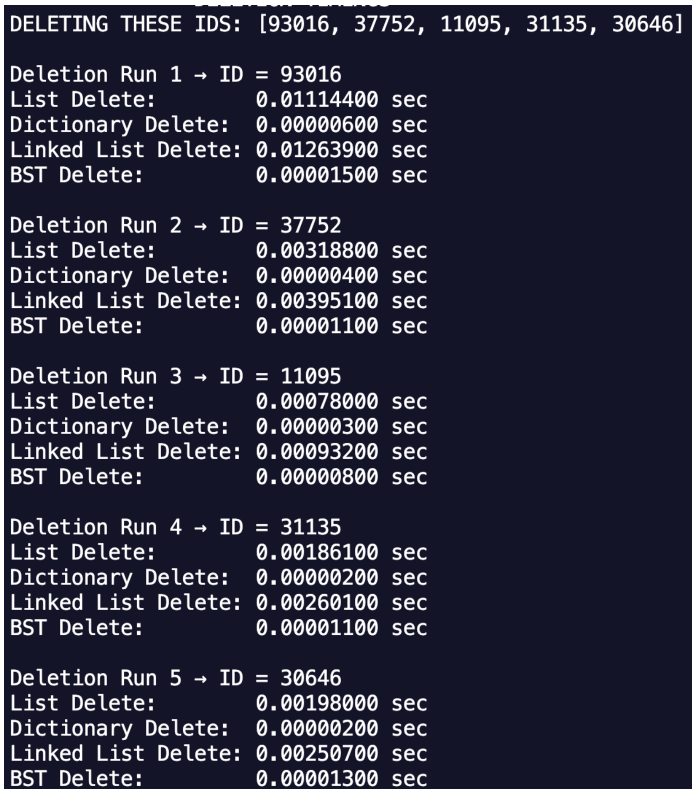


```
Insertion Times:

Insertion Run 1  →  ID = 100010
List Insert:        0.00000500 sec
Dictionary Insert:  0.00000300 sec
Linked List Insert: 0.00000200 sec
BST Insert:         0.00000300 sec

Insertion Run 2  →  ID = 100020
List Insert:        0.00000200 sec
Dictionary Insert:  0.00000200 sec
Linked List Insert: 0.00000200 sec
BST Insert:         0.00000100 sec

Insertion Run 3  →  ID = 100030
List Insert:        0.00000000 sec
Dictionary Insert:  0.00000100 sec
Linked List Insert: 0.00000000 sec
BST Insert:         0.00000100 sec

Insertion Run 4  →  ID = 100040
List Insert:        0.00000000 sec
Dictionary Insert:  0.00000000 sec
Linked List Insert: 0.00000100 sec
BST Insert:         0.00000200 sec

Insertion Run 5  →  ID = 100050
List Insert:        0.00000100 sec
Dictionary Insert:  0.00000000 sec
Linked List Insert: 0.00000000 sec
BST Insert:         0.00000200 sec
```

4. Deletion.

List	Dictionaries	Binary Search Tree	Linked List
0.01114400 sec.	0.00000600 sec.	0.00001500 sec.	0.01263900 sec.
0.00318800 sec.	0.00000400 sec.	0.00001100 sec.	0.00395100 sec.
0.00078000 sec.	0.00000300 sec.	0.00000800 sec.	0.00260100 sec.
0.00186100 sec.	0.00000200 sec.	0.00001100 sec.	0.00093200 sec.
0.00198000 sec.	0.00000200 sec.	0.00001300 sec.	0.00250700 sec.

Screenshot 7: Deletions



```
DELETING THESE IDS: [93016, 37752, 11095, 31135, 30646]

Deletion Run 1 → ID = 93016
List Delete:        0.01114400 sec
Dictionary Delete:  0.00000600 sec
Linked List Delete: 0.01263900 sec
BST Delete:         0.00001500 sec

Deletion Run 2 → ID = 37752
List Delete:        0.00318800 sec
Dictionary Delete:  0.00000400 sec
Linked List Delete: 0.00395100 sec
BST Delete:         0.00001100 sec

Deletion Run 3 → ID = 11095
List Delete:        0.00078000 sec
Dictionary Delete:  0.00000300 sec
Linked List Delete: 0.00093200 sec
BST Delete:         0.00000800 sec

Deletion Run 4 → ID = 31135
List Delete:        0.00186100 sec
Dictionary Delete:  0.00000200 sec
Linked List Delete: 0.00260100 sec
BST Delete:         0.00001100 sec

Deletion Run 5 → ID = 30646
List Delete:        0.00198000 sec
Dictionary Delete:  0.00000200 sec
Linked List Delete: 0.00250700 sec
BST Delete:         0.00001300 sec
```

Zoom Link:

https://psu.zoom.us/rec/share/xbufEd2S9Joxja-s3-in7yjmIwVXz1YSetM2cvA95mVZ4xuXjWK4ICPIwc0q8S0K.pzC_xTYej-rLgdqv?startTime=1763862582000