

# Searching in Linked Lists and Arrays

## 1. Searching in Arrays

To calculate the Time Complexity:

1. Creating a random list of integers with 'n' items between '1' and 'm'. Since 'n' iterations are required to produce a set of random numbers, this procedure requires  $O(n)$  time.
2. Counting the number of values larger than 50: This operation iterates through the full list once to count the values greater than 50, and in the worst case, it takes  $O(n)$  time.
3. Sorting the list using merge sort: The merge sort method, which splits the list in half recursively and then combines the results, has an average and worst-case time complexity of  $O(n \log n)$ . Therefore, it takes  $O(n \log n)$  time to sort the list.
4. Removing an element from the list: Removing an element from a list at a certain index may require moving elements after the deleted element, hence in the worst case, doing so will take  $O(n)$  time.
5. Adding an element to the list: In the worst situation, adding an element to a list at a certain location requires moving members beyond the insertion point, which adds additional time that is  $O(n)$  in size.

The overall time complexity of the `searcharray` function can be approximated as follows:

1. If there are more than 5 values greater than 50:

- Sorting the list:  $O(n \log n)$
- Deleting an element:  $O(n)$
- Inserting an element:  $O(n)$
- Total time complexity:  $O(n \log n)$

2. If there are less than 5 values greater than 50:

- Sorting the list in descending order:  $O(n \log n)$
- Deleting an element:  $O(n)$
- Inserting an element:  $O(n)$
- Total time complexity:  $O(n \log n)$

In summary, the code's overall time complexity is  $O(n \log n)$  because the most time-consuming operation is the sorting of the list using the merge sort algorithm.

### Number of operations/instructions:

1.  $F(n) = 10n + (n \cdot \log n) + 21$
2. Neglecting constant 21
3.  $F(n) = 10n + (n \cdot \log n)$
4. Since  $n \cdot \log n$  has higher order than  $10n$
5. Asymptotically  $F(n)$  can be represented as:  
$$F(n) = O(n \cdot \log n)$$

To calculate the Space Complexity:

1. **Input Space:** Since the space needed to store user inputs 'n' and 'm' is small and constant, it has little impact on the complexity of the space.
2. **List 'result':** The 'result' list, which contains 'n' randomly generated integers, is the main factor affecting the space complexity. As a result, this list's space complexity is  $O(n)$ .
3. **Additional Variables:** Several more variables, including "mid," "left," "right," "left index," "right index," "result," "count," and "loopcounter," are utilized within the functions. These variables do not increase in size with the input. As a result, their space complexity is constant and  $O(1)$ .
4. **Randomly Generated Data:** the 'random.randint(1, m)' function generates random integers and inserts them into the linked list. The space complexity for the randomly generated data depends on the number of elements generated, which is  $O(n)$ .
5. **A recursive implementation is used for the "merge\_sort" function.** The list is continually divided in half; therefore, in the result, the space complexity brought on by the recursion may be calculated as  $O(\log n)$ .

In conclusion, the 'result' list, which has an  $O(n)$  space complexity, is the main contributor to the space complexity. Other variables and recursive calls have comparatively low space complexity, which is constant. Because the 'result' list requires space, the code's overall space complexity is  $O(n)$ .

Note: Learnt and analyzed how to calculate the time complexity from google and ChatGPT and implemented the same.

### Program Run:

1. n= 50  
m=100  
time taken = 0.0 microseconds (with print statements)  
space taken = 472 bytes

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS W:\WMU Assignments\Program for Grad\Assignment 1> & C:/Users/rohan/AppData/Local/Microsoft/WindowsApps/python3.11.exe "w:\WMU Assignments\Program for Grad\Assignment 1\searcharray_norep.py"
Enter the number of integers n: 50
Enter the ending range of integers m: 100
[46, 98, 97, 91, 19, 11, 84, 99, 8, 47, 82, 35, 41, 52, 100, 49, 78, 78, 54, 2, 51, 73, 57, 46, 56, 51, 77, 1, 22, 92, 9, 56, 85, 36, 79, 36, 86, 35, 77, 33, 87, 18, 67, 26, 41, 36, 18, 57, 67, 26]
There are more than 5 values Greater than 50
Sorting in Ascending order
[1, 2, 8, 9, 11, 18, 18, 19, 22, 26, 26, 33, 35, 35, 36, 36, 36, 41, 41, 46, 46, 47, 49, 51, 51, 52, 54, 56, 56, 57, 57, 67, 67, 73, 77, 77, 78, 78, 79, 82, 84, 85, 86, 87, 91, 92, 97, 98, 99, 100]
Deleting the 5th Element
[1, 2, 8, 9, 18, 18, 19, 22, 26, 26, 33, 35, 35, 36, 36, 41, 41, 46, 46, 47, 49, 51, 51, 52, 54, 56, 56, 57, 57, 67, 67, 73, 77, 77, 78, 78, 79, 82, 84, 85, 86, 87, 91, 92, 97, 98, 99, 100]
The final array is
[1, 2, 8, 9, 18, 18, 18, 19, 22, 26, 26, 33, 35, 35, 36, 36, 36, 41, 41, 46, 46, 47, 49, 51, 51, 52, 54, 56, 56, 57, 57, 67, 67, 73, 77, 77, 78, 78, 79, 82, 84, 85, 86, 87, 91, 92, 97, 98, 99, 100]
Total Time taken 0 microseconds
Space taken is 472 bytes
PS W:\WMU Assignments\Program for Grad\Assignment 1>

```

2.  $n = 100$

m = 200

time taken = 0.0 microseconds (with print statements)

space taken = 920 bytes

```
PS W:\WMU Assignments\Program for Grad\Assignment 1> & C:/Users/rohan/AppData/Local/Microsoft/WindowsApps/python3.11.exe "w:/WMU Assignments/Program for Grad/Assignment 1/se
archarray_norep.py"
Enter the number of integers n: 100
Enter the ending range of integers m: 200
[41, 39, 8, 175, 172, 183, 181, 81, 90, 114, 5, 150, 129, 38, 81, 10, 133, 50, 25, 17, 56, 109, 149, 157, 122, 7, 119, 9, 151, 66, 77, 112, 11, 14, 163, 19, 48, 25, 15, 69,
57, 70, 164, 6, 127, 38, 176, 181, 50, 167, 164, 164, 186, 180, 173, 199, 72, 115, 3, 4, 105, 92, 186, 86, 100, 129, 90, 129, 138, 105, 111, 48, 152, 168, 194, 9, 72, 42, 11
1, 118, 151, 187, 105, 15, 104, 11, 72, 155, 16, 42, 120, 109, 46, 134, 22, 167, 200, 165, 79, 166]
There are more than 5 values Greater than 50
Sorting in Ascending order
[3, 4, 5, 6, 7, 8, 9, 9, 10, 11, 11, 14, 15, 15, 16, 17, 19, 22, 25, 25, 38, 38, 39, 41, 42, 42, 46, 48, 48, 50, 50, 56, 57, 66, 69, 70, 72, 72, 72, 77, 79, 81, 81, 86, 90,
90, 92, 100, 104, 105, 105, 105, 109, 109, 111, 111, 112, 114, 115, 118, 119, 120, 122, 127, 129, 129, 129, 133, 134, 138, 149, 150, 151, 151, 152, 155, 157, 163, 164, 164,
164, 165, 166, 167, 167, 168, 172, 173, 175, 176, 180, 181, 181, 183, 186, 186, 187, 194, 199, 200]
Deleting the 5th Element
[3, 4, 5, 6, 8, 9, 9, 10, 11, 11, 14, 15, 15, 16, 17, 19, 22, 25, 25, 38, 38, 39, 41, 42, 42, 46, 48, 48, 50, 50, 56, 57, 66, 69, 70, 72, 72, 72, 77, 79, 81, 81, 86, 90, 90,
92, 100, 104, 105, 105, 105, 109, 109, 111, 111, 112, 114, 115, 118, 119, 120, 122, 127, 129, 129, 129, 133, 134, 138, 149, 150, 151, 151, 152, 155, 157, 163, 164, 164, 164,
165, 166, 167, 167, 168, 172, 173, 175, 176, 180, 181, 181, 183, 186, 186, 187, 194, 199, 200]
The final array is
[3, 4, 5, 6, 8, 9, 9, 10, 10, 11, 11, 14, 15, 15, 16, 17, 19, 22, 25, 25, 38, 38, 39, 41, 42, 42, 46, 48, 48, 50, 50, 56, 57, 66, 69, 70, 72, 72, 72, 77, 79, 81, 81, 86, 90,
90, 92, 100, 104, 105, 105, 105, 109, 109, 111, 111, 112, 114, 115, 118, 119, 120, 122, 127, 129, 129, 129, 133, 134, 138, 149, 150, 151, 151, 152, 155, 157, 163, 164, 164,
164, 165, 166, 167, 167, 168, 172, 173, 175, 176, 180, 181, 181, 183, 186, 186, 187, 194, 199, 200]
Total Time taken 0 microseconds
Space taken is 920 bytes
PS W:\WMU Assignments\Program for Grad\Assignment 1> []
```

3. n = 500

m = 300

r=10

Average time taken = 1501 microseconds ( without print statements )

space taken = 4216 bytes

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS W:\WMU Assignments\Program for Grad\Assignment 1> & C:/Users/rohan/AppData/Local/Microsoft/WindowsApps/python3.11.exe "w:/WMU Assignments/Program for Grad/Assignment 1/se
archarray_withrep.py"
Enter the number of integers n: 500
Enter the ending range of integers m: 300
Enter the value of repetitions r: 10
Times for 10 repetitions is [0, 0, 0, 0, 0, 0, 0, 15010, 0, 0]
Average time 1501 microseconds
Space taken is 4216 bytes
PS W:\WMU Assignments\Program for Grad\Assignment 1> []
```

4. n = 1000

m = 5000

r = 10

Average time taken for 10 Recursion = 3569 microseconds ( without print statements )

Space taken = 8856 bytes

```
PS W:\WMU Assignments\Program for Grad\Assignment 1> & C:/Users/rohan/AppData/Local/Microsoft/WindowsApps/python3.11.exe "w:/WMU Assignments/Program for Grad/Assignment 1/se
archarray_withrep.py"
Enter the number of integers n: 1000
Enter the ending range of integers m: 5000
Enter the value of repetitions r: 10
Times for 10 repetitions is [0, 0, 15529, 0, 0, 0, 0, 16561, 3600]
Average time 3569 microseconds
Space taken is 8856 bytes
PS W:\WMU Assignments\Program for Grad\Assignment 1> []
```

5. n = 50000

m = 10000

r = 10

Average time taken for 10 Recursion = 193110 microseconds (without print statements)

Space taken = 444376 bytes

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Python + - [ ] [ ] ... ^ >
PS W:\WMU Assignments\Program for Grad\Assignment 1> & C:\Users\rohan\AppData\Local\Microsoft\WindowsApps\python3.11.exe "w:\WMU Assignments\Program for Grad\Assignment 1\se
archarray_withrep.py"
Enter the number of integers n: 50000
Enter the ending range of integers m: 10000
Enter the value of repetitions r: 10
Times for 10 repetitions is [189337, 187500, 203126, 189608, 187456, 204610, 187531, 187544, 205647, 188745]
Average time 193110 microseconds
Space taken is 444376 bytes
PS W:\WMU Assignments\Program for Grad\Assignment 1> [ ]

```


6.  $n = 100000$

$m = 50000$

$$r = 7$$

Average time taken for 7 Recursion = 406195 microseconds ( without print statements)

Space taken = 800984 bytes



```
PS W:\WMU Assignments\Program for Grad\Assignment 1> & C:/Users/rohan/AppData/Local/Microsoft/WindowsApps/python3.11.exe "w:\WMU Assignments\Program for Grad\Assignment 1/se
archarray_withrep.py"
Enter the number of integers n: 100000
Enter the ending range of integers m: 50000
Enter the value of repetitions r: 7
Times for 7 repetitions is [383746, 390625, 408409, 405286, 440299, 407426, 407574]
Average time 406195 microseconds
Space taken is 800984 bytes
PS W:\WMU Assignments\Program for Grad\Assignment 1>
```

**Graph:** (Graph is plotted in a separate file “time\_space\_complexity\_graph.xlsx”)

## 1. Time Complexity Graph

Values of n	Average Time in microseconds
50	0
100	0
500	1501
1000	3569
50000	193110
100000	406195

The graph illustrates the time complexity of an array operation. The x-axis represents 'N VALUES' from 0 to 120,000, and the y-axis represents 'AVERAGE TIME IN MICROSECONDS' from 0 to 450,000. The data points show a linear increase in time as the number of values increases, confirming the  $O(N)$  time complexity.

N Values	Average Time in Microseconds
50	0
100	0
500	1501
1000	3569
50000	193110
100000	406195

## 2. Space Complex Graph

Values of n	Space taken in bytes
50	472
100	920
500	4216
1000	8856
50000	444376
100000	800984

**SPACE COMPLEXITY OF ARRAY**

The graph illustrates the linear relationship between the number of elements (N) and the space required. The data points are as follows:

N VALUES	SPACE IN BYTES
50	472
100	920
500	4216
1000	8856
50000	444376
100000	800984

## 2. Searching in Linked Lists

To calculate the Time Complexity:

### Theoretical Analysis:

1. Creating the Doubly Linked List and inserting 'n' random integers. Hence, its  $O(n)$
2. Counting the number of values greater than 50:
  - The code iterates over the linked list to count the number of values greater than 50. It stops when it finds 5 such values.
  - The worst-case scenario is that all values are greater than 50, so it could potentially iterate over all 'n' elements.
3. Sorting the linked list: The method uses a quicksort algorithm, which has an average time complexity of  $O(n \log n)$ .
4. Deleting an element: sometimes it's either 5<sup>th</sup> element or 2<sup>nd</sup> element based on the sorting order. So, it depends on the input 'n' Hence the worst-case scenario is  $O(n)$
5. Inserting a new element: The insert\_sorted method inserts an element in its correct position, which is also an  $O(n)$  operation in the worst case.
6. Printing the linked list: Printing the linked list involves iterating over all 'n' elements once. Hence, its  $O(n)$

Overall, the time complexity of the code depends significantly on the sorting operation. In the average case, it is  $O(n * \log(n))$  due to Quick Sort.

### Number of operations/instructions:

6.  $F(n) = 34n + (n * \log n) + 47$
7. Neglecting constant 47
8.  $F(n) = 34n + n * \log n$
9. Since  $n * \log n$  has higher order than  $34n$
10. Asymptotically  $F(n)$  can be represented as:  
$$F(n) = O(n * \log n)$$

To Calculate the space complexity:

### 1. Doubly Linked List:

- The doubly linked list is the main data structure in this code. It is made up of nodes, each of which contains data and includes references to the nodes before and after it.
- The doubly linked list has an  $O(n)$  space complexity, where  $n$  is the number of elements added to the list.

### 2. Additional Variables:

- Many variables are used to store temporary values, including 'new\_node', 'current', 'last\_node', 'pivot', 'start', 'end', and 'result'.
- The space required for these variables is  $O(1)$  because the number of these variables does not depend on the size of the input.

### 3. Randomly Generated Data:

- the 'random.randint(1, m)' function generates random integers and inserts 'n' times into the linked list. These integers occupy additional space in memory.

- The space complexity for the randomly generated data depends on the number of elements generated, which is  $O(n)$ .

Overall, the space complexity of the code is  $O(n)$ , where  $n$  is the number of elements in the linked list.

Note: Learnt and analyzed how to calculate the time complexity from google and chatgpt and implemented the same.

## Program Run:

1.  $n=50$   
 $m=100$   
time taken = 0.0 microseconds (with print statements)  
space taken = 2800 bytes

```
PS W:\WMU Assignments\Program for Grad\Assignment 1> C:\Users\rohan\AppData\Local\Microsoft\WindowsApps\python3.11.exe "w:
/WMU Assignments/Program for Grad/Assignment 1/searchlinkedlist_norep.py"
Enter the number of integers n: 50
Enter the ending range of integers m: 100
70->95->29->84->21->49->97->15->42->54->24->12->46->54->78->32->36->81->3->85->65->92->11->35->47->27->45->53->49->62->46->6
0->34->79->14->46->19->99->51->60->92->98->28->98->80->38->53->24->83->83->None
There are more than 5 values Greater than 50
Sorting in Ascending order
3->11->12->14->15->19->21->24->24->27->28->29->32->34->35->36->38->42->45->46->46->46->47->49->49->51->53->53->54->54->60->6
0->62->65->70->78->79->80->81->83->83->84->85->92->92->95->97->98->98->99->None
Deleting the 5th Element
3->11->12->14->19->21->24->27->28->29->32->34->35->36->38->42->45->46->46->47->49->49->51->53->53->54->54->60->60->6
2->65->70->78->79->80->81->83->83->84->85->92->92->95->97->98->98->99->None
Inserted 10 in its correct place
3->10->11->12->14->19->21->24->24->27->28->29->32->34->35->36->38->42->45->46->46->47->49->49->51->53->53->54->54->60->6
0->62->65->70->78->79->80->81->83->83->84->85->92->92->95->97->98->98->99->None
Total time taken is 0 microseconds
Space taken is 2800 bytes
PS W:\WMU Assignments\Program for Grad\Assignment 1> █
```

2.  $n = 100$   
 $m = 200$   
time taken = 0.0 microseconds (with print statements)  
space taken = 5600 bytes

```
PS W:\WMU Assignments\Program for Grad\Assignment 1> C:\Users\rohan\AppData\Local\Microsoft\WindowsApps\python3.11.exe "w:/WMU Assignments/Program for Grad
/Assignment 1/searchlinkedlist_norep.py"
Enter the number of integers n: 100
Enter the ending range of integers m: 200
44->60->111->7->126->130->67->73->20->149->4->105->77->30->173->70->16->179->47->128->146->7->191->105->124->84->150->92->116->51->55->100->59->183->157->178
->12->32->157->107->127->89->28->165->163->191->176->106->161->133->52->192->4->179->171->106->27->59->160->122->95->172->67->184->160->69->168->180->47->61-
>88->182->70->15->85->5->92->118->45->55->84->95->163->57->37->172->136->30->185->111->115->35->67->73->34->137->8->46->137->88->None
There are more than 5 values Greater than 50
Sorting in Ascending order
4->4->5->7->7->8->12->15->16->20->27->28->30->30->32->34->35->37->44->45->46->47->47->51->52->55->55->57->59->59->60->61->67->67->67->69->70->70->73->73->77-
>84->84->85->88->88->89->92->92->95->95->100->105->105->106->106->107->111->111->115->116->118->122->124->126->127->128->130->133->136->137->137->146->149->1
50->157->157->160->160->161->163->163->165->168->171->172->172->173->176->178->179->179->180->182->183->184->185->191->191->192->None
Deleting the 5th Element
4->4->5->7->8->12->15->16->20->27->28->30->30->32->34->35->37->44->45->46->47->47->51->52->55->55->57->59->59->60->61->67->67->67->69->70->70->73->73->77-
>84->85->88->88->89->92->92->95->95->100->105->105->106->106->107->111->111->115->116->118->122->124->126->127->128->130->133->136->137->137->146->149->150-
>157->157->160->160->161->163->163->165->168->171->172->172->173->176->178->179->179->180->182->183->184->185->191->191->192->None
Inserted 10 in its correct place
4->4->5->7->8->10->12->15->16->20->27->28->30->30->32->34->35->37->44->45->46->47->47->51->52->55->55->57->59->59->60->61->67->67->67->69->70->70->73->73->77-
>84->84->85->88->88->89->92->92->95->95->100->105->105->106->106->107->111->111->115->116->118->122->124->126->127->128->130->133->136->137->137->146->149->
150->157->157->160->160->161->163->163->165->168->171->172->172->173->176->178->179->179->180->182->183->184->185->191->191->192->None
Total time taken is 0 microseconds
Space taken is 5600 bytes
PS W:\WMU Assignments\Program for Grad\Assignment 1> █
```

3. n = 500

m = 300

r=10

Average time taken for 10 Recursion = 1553 microseconds ( without print statements )

Space taken = 28000 bytes

```
PS W:\WMU Assignments\Program for Grad\Assignment 1> & C:/Users/rohan/AppData/Local/Microsoft/WindowsApps/python3.11.exe "w:\WMU Assignments\Program for Grad
/Assignment 1/searchlinkedlist_withrep.py"
Enter the number of integers n: 500
Enter the ending range of integers m: 300
Enter the value of repetitions r: 10
Times for 10 repetitions is [0, 0, 15534, 0, 0, 0, 0, 0, 0, 0]
Average time 1553 microseconds
Space taken is 28000 bytes
PS W:\WMU Assignments\Program for Grad\Assignment 1> 
```

4. n = 1000

m = 5000

r = 10

Average time taken for 10 Recursion = 3114 microseconds ( without print statements )

Space taken = 56000 bytes

```
PS W:\WMU Assignments\Program for Grad\Assignment 1> & C:/Users/rohan/AppData/Local/Microsoft/WindowsApps/python3.11.exe "w:\WMU Assignments\Program for Grad
/Assignment 1/searchlinkedlist_withrep.py"
Enter the number of integers n: 1000
Enter the ending range of integers m: 5000
Enter the value of repetitions r: 10
Times for 10 repetitions is [0, 15519, 0, 0, 0, 0, 0, 0, 0, 15622]
Average time 3114 microseconds
Space taken is 56000 bytes
PS W:\WMU Assignments\Program for Grad\Assignment 1> 
```

5. n = 50000

m = 10000

r = 10

Average time taken for 10 Recursion = 158140 microseconds (without print statements)

Space taken = 2800000 bytes

```
PS W:\WMU Assignments\Program for Grad\Assignment 1> & C:/Users/rohan/AppData/Local/Microsoft/WindowsApps/python3.11.exe "w:\WMU Assignments\Program for Grad
/Assignment 1/searchlinkedlist_withrep.py"
Enter the number of integers n: 50000
Enter the ending range of integers m: 10000
Enter the value of repetitions r: 10
Times for 10 repetitions is [148507, 156246, 156249, 165994, 140616, 171554, 143946, 165725, 156262, 176301]
Average time 158140 microseconds
Space taken is 2800000 bytes
PS W:\WMU Assignments\Program for Grad\Assignment 1> 
```

6. n = 100000

m = 50000

r = 7

Average time taken for 7 Recursion = 323016 microseconds ( without print statements)

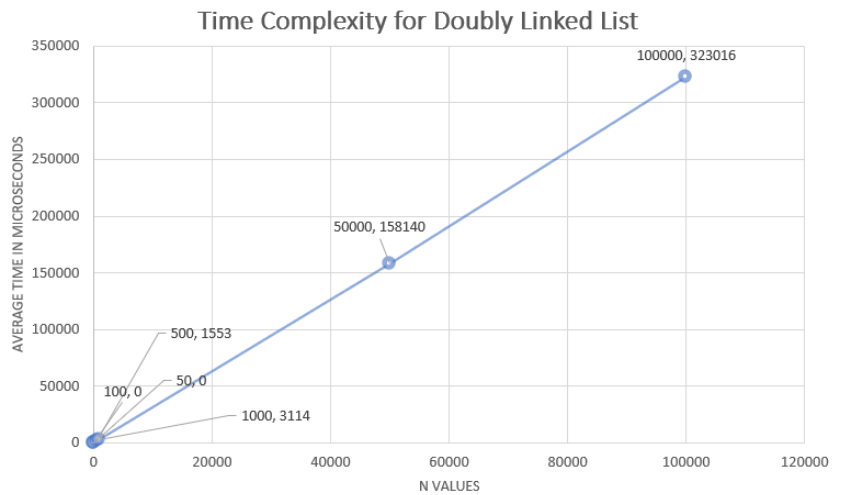
Space taken = 5600000 bytes

```
PS W:\WMU Assignments\Program for Grad\Assignment 1> & C:/Users/rohan/AppData/Local/Microsoft/WindowsApps/python3.11.exe "w:\WMU Assignments\Program for Grad
/Assignment 1/searchlinkedlist_withrep.py"
Enter the number of integers n: 100000
Enter the ending range of integers m: 50000
Enter the value of repetitions r: 7
Times for 7 repetitions is [328557, 313157, 330250, 320684, 310613, 346200, 311651]
Average time 323016 microseconds
Space taken is 5600000 bytes
PS W:\WMU Assignments\Program for Grad\Assignment 1> 
```

**Graph:** (Graph is plotted in a separate file “time\_space\_complexity\_graph.xlsx”)

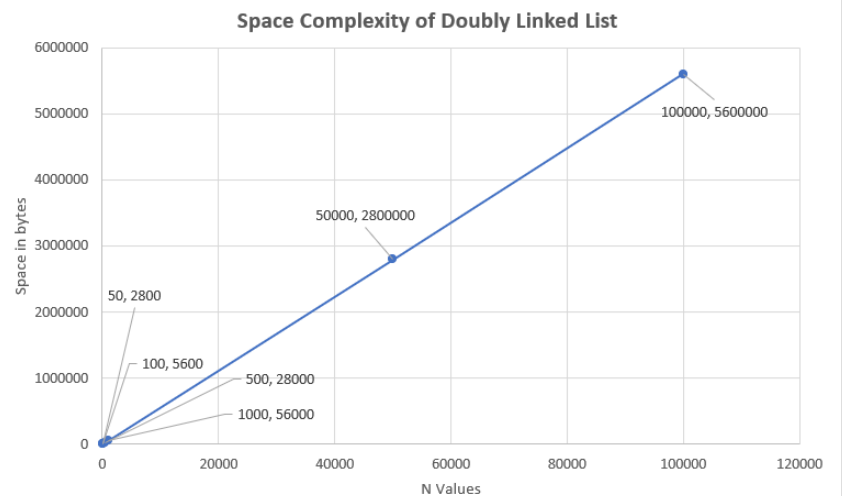
### 1. Time Complexity for Doubly Linked List

Values of n	Average Time in microseconds
50	0
100	0
500	1553
1000	3114
50000	158140
100000	323016



### 2. Space Complexity for Doubly Linked List

Values of n	Space in bytes
50	2800
100	5600
500	28000
1000	56000
50000	2800000
100000	5600000

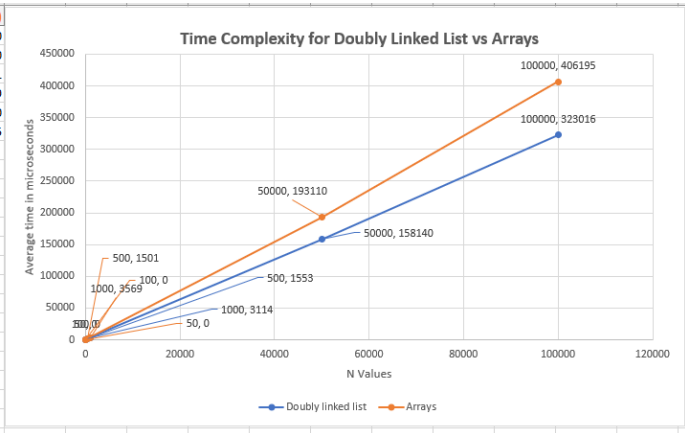




## Linked List vs Arrays:

### 1. Time Complexity Comparison

Values of n	Average Time in microseconds (DLL)	Average Time in microseconds (Array)
50	0	0
100	0	0
500	1553	1501
1000	3114	3569
50000	158140	193110
100000	323016	406195



### 2. Space Complexity Comparison

Values of n	Space in Bytes (DLL)	Space in Bytes (Array)
50	2800	472
100	5600	920
500	28000	4216
1000	56000	8856
50000	2800000	444376
100000	5600000	800984

