TIME COMPLEXITY ANALYSIS

METHOD	SINGLY LINKED LIST	DYNAMIC ARRAY
Insert at index	O(N)	O(N)
Delete at index	O(N)	O(N)
Returns size	O(N)	0(1)
Empty or not	0(1)	0(1)
Rotate	O(n+k)	0(k)
Reverse	O(N)	O(N)
Append at End	O(N)	O(N)
Prepends at Beginning	0(1)	O(N)
Merge	O(n+m)	O(n+m)
Interleaves	0(n+m)	O(n+m)
Get middle element	O(N)	0(1)
Index of element	O(N)	O(N)
Splits at Index	O(N)	O(N)

SPACE COMPLEXITY ANALYSIS

OPERATION	SINGLY LINKED LIST	DYNAMIC ARRAY
Memory Overhead	O(N)	O(N)
Space Utilization	May have unused space due to overallocatio	No unused space beyond node pointers

SINGIY LINKED LIST

- **Advantages :-**

- It is very easier for the accessibility of a node in the forward direction.
- The insertion and deletion of a node are very easy.
- The Requirement will less memory when compared to doubly, circular or doubly circular linked list.
- The Singly linked list is the very easy data structure to implement.
- During the execution, we can allocate or deallocate memory easily.

- **Disadvantages :-**

- Accessing the preceding node of a current node is not possible as there is no backward traversal.
- The Accessing of a node is very time-consuming.

DYNAMIC ARRAY

- **Advantages:-**

- Efficient memory usage: Dynamic arrays can grow or shrink in size as needed, reducing the amount of wasted memory compared to fixed-size arrays.
- Easy to use: Dynamic arrays can be implemented using built-in data structures like vectors in C++ or ArrayLists in Java, making them easy to use for developers.
- Flexible: Dynamic arrays can be used to store any type of data, including primitive types and objects.

- **Disadvantages:-**

- Less efficient than arrays for some operations: Dynamic arrays can be less efficient than arrays for certain operations, such as inserting elements at the beginning or in the middle of the array.
- Memory overhead: Dynamic arrays can have a small amount of memory overhead due to the need to keep track of the array's size and capacity.

CREATE A REPORT ON COMPARISON OF LINKED LIST AND DYNAMIC ARRAY

DYNAMIC ARRAY	SINGLY LINKED LISTS
1. Arrays are stored in contiguous location.	1. Linked lists are not stored in contiguous location.
2. Fixed in size.	2. Dynamic in size.
3. Memory is allocated at compile time.	3. Memory is allocated at run time.
4. Uses less memory than linked lists.	4. Uses more memory because it stores both data and the address of next node.

5. Elements can be accessed easily.	5. Element accessing requires the traversal of whole linked list.
6. Insertion and deletion operation takes time.	6. Insertion and deletion operation is faster.

- **Uses Singly Linked Lists :-**

- Frequent insertion and deletion of elements are required.
- Memory efficiency is crucial.

- **Use Dynamic Arrays when :-**

- Fast search is required.
- Cache-friendly access is important.
- Easy implementation is desired.

CONCLUSION

In conclusion, Singly Linked Lists and Dynamic Arrays are both useful data structures, each with their own strengths and weaknesses. Singly Linked Lists are suitable for applications that require frequent insertion and deletion of elements, while Dynamic Arrays are better suited for applications that require fast search and cache-friendly access. When choosing between the two, consider the specific requirements of the problem and the trade-offs between insertion/deletion efficiency, search efficiency, and memory usage.