Compare the following aspects of linked lists and dynamic arrays:

o Time complexity of each method

o Space complexity of each method

o Advantages and disadvantages of each data structure

Solution :

Time Complexity of each method :

|  |  |  |
| --- | --- | --- |
| Operations | Linked List | Dynamic Array |
| Accessing element by index | O(n) : Requires Traversing From the head (or tail,in the case of DLL) to the desired index. | O(1) :Direct access elements using indexing |
| Insertion at beginning | O(1) : Simple updation of pointer | O(n) :shifting all elements to the right |
| Insertion at end | O(1) : Updating tail pointer | O(1) :Constant time on average due to occasional resizing |
| Insertion at a specific index | O(n) : Traversing to the index | O(n) :shifting elements makes space |
| Deletion at beginning | O(1) : Simple updation of pointer | O(n) :shifting all elements to left |
| Deletion at end | O(n) :Traversing to second-to-last node to update tail pointer | O(1) : Constant time on average due to occasional resizing |
| Deletion at a specific index | O(n) :Traversing to the index | O(n) :Requires shifting elements after the deleted index. |

Space Complexity of each method :

* Linked List :

Space complexity for storing elements :O(n) - Requires additional memory for each node.

* Dynamic arrays :

Space complexity for storing elements : O(n) - Requires contiguous memory allocation for elements.

Advantages and Disadvantages of each data structures :

Linked List

|  |  |
| --- | --- |
| Advantages | Disadvantages |
| Dynamic size : Dynamically grow & shrink in size without need for resizing. | Inefficient random access : Accessing of elements by index requires traversing list from head(or tail) |
| Efficient insertion & deletion at beginning : This operations are constant time | Extra memory overhead : Requires additional memory for storing pointers. |
| No need for continguous memory : Each node can be allocated memory at different locations . | Cache inefficiency : Poor cache locality due to scattered memory locations . |

Dynamic Arrays

|  |  |
| --- | --- |
| Advantages | Dis-Advantages |
| Efficient random access : Elements can be accessed directly using array indexing. | Fixed size : Dynamic arrays need resizing when they reach capacity,which can be costly. |
| Better cache locality : Elements are stored contiguously, which improves cache efficiency. | Costly insertion & deletion in middle :Requires shifting elements to maintain contiguous memory. |
| Lower memory overhead : No extra memory needed for pointers/ references. | Costly re-sizing : re-sizing operations may require allocating new memory & copying elements ,leading to occasional performance degradation. |

At last , Linked lists are more suitable when frequent insertions & deletions at the beginning are required , and memory overhead is concern.

Dynamic arrays , on the other hand , excel in scenarios where efficient random access & better cache locality are essential, and the collection size is relatively stable or predictable.