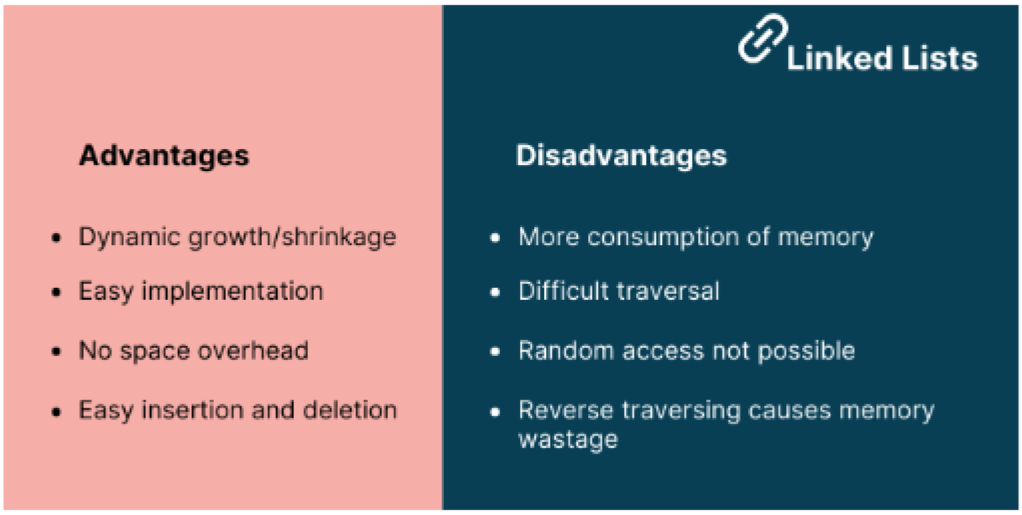

Linked Lists

Advantages

- Dynamic growth/shrinkage
- Easy implementation
- No space overhead
- Easy insertion and deletion

Disadvantages

- More consumption of memory
- Difficult traversal
- Random access not possible
- Reverse traversing causes memory wastage

Advantages Of Linked List:

- **Dynamic data structure:** A linked list is a dynamic arrangement so it can grow and shrink at runtime by allocating and deallocating memory. So there is no need to give the initial size of the linked list.
- **No memory wastage:** In the Linked list, efficient memory utilization can be achieved since the size of the linked list increase or decrease at run time so there is no memory wastage and there is no need to pre-allocate the memory.
- **Implementation:** Linear data structures like stacks and queues are often easily implemented using a linked list.
- **Insertion and Deletion Operations:** Insertion and deletion operations are quite easier in the linked list. There is no need to shift elements after the insertion or deletion of an element only the address present in the next pointer needs to be updated.
- **Flexible:** This is because the elements in Linked List are not stored in contiguous memory locations unlike the array.
- **Efficient for large data:** When working with large datasets linked lists play a crucial role as it can grow and shrink dynamically.
- **Scalability:** Contains the ability to add or remove elements at any position.

Disadvantages Of Linked List:

- **Memory usage:** More memory is required in the linked list as compared to an array. Because in a linked list, a pointer is also required to store the address of the next element and it requires extra memory for itself.
- **Traversal:** In a Linked list traversal is more time-consuming as compared to an array. Direct access to an element is not possible in a linked list as in an array by index. For example, for accessing a node at position n, one has to traverse all the nodes before it.
- **Reverse Traversing:** In a singly linked list reverse traversing is not possible, but in the case of a doubly-linked list, it can be possible as it contains a pointer to the previously connected nodes with each node. For performing this extra memory is required for the back pointer hence, there is a wastage of memory.

- **Random Access:** Random access is not possible in a linked list due to its dynamic memory allocation.
- **Lower efficiency at times:** For certain operations, such as searching for an element or iterating through the list, can be slower in a linked list.
- **Complex implementation:** The linked list implementation is more complex when compared to array. It requires a complex programming understanding.
- **Difficult to share data:** This is because it is not possible to directly access the memory address of an element in a linked list.
- **Not suited for small dataset:** Cannot provide any significant benefits on small dataset compare to that of an array.

Advantages of Stack:

- **Easy implementation:** Stack data structure is easy to implement using arrays or linked lists, and its operations are simple to understand and implement.
- **Efficient memory utilization**: Stack uses a contiguous block of memory, making it more efficient in memory utilization as compared to other data structures.
- **Fast access time:** Stack data structure provides fast access time for adding and removing elements as the elements are added and removed from the top of the stack.
- **Helps in function calls:** Stack data structure is used to store function calls and their states, which helps in the efficient implementation of recursive function calls.
- **Supports backtracking:** Stack data structure supports backtracking algorithms, which are used in problem-solving to explore all possible solutions by storing the previous states.
- **Used in Compiler Design:** Stack data structure is used in compiler design for parsing and syntax analysis of programming languages.
- **Enables undo/redo operations**: Stack data structure is used to enable undo and redo operations in various applications like text editors, graphic design tools, and software development environments.
-

Disadvantages of Stack:

- **Limited capacity:** Stack data structure has a limited capacity as it can only hold a fixed number of elements. If the stack becomes full, adding new elements may result in stack overflow, leading to the loss of data.
- **No random access:** Stack data structure does not allow for random access to its elements, and it only allows for adding and removing elements from the top of the stack. To access an element in the middle of the stack, all the elements above it must be removed.
- **Memory management:** Stack data structure uses a contiguous block of memory, which can result in memory fragmentation if elements are added and removed frequently.

- **Not suitable for certain applications:** Stack data structure is not suitable for applications that require accessing elements in the middle of the stack, like searching or sorting algorithms.
- **Stack overflow and underflow**: Stack data structure can result in stack overflow if too many elements are pushed onto the stack, and it can result in stack underflow if too many elements are popped from the stack.
- **Recursive function calls limitations:** While stack data structure supports recursive function calls, too many recursive function calls can lead to stack overflow, resulting in the termination of the program.

Advantages of Queue:

- A large amount of data can be managed efficiently with ease.
- Operations such as insertion and deletion can be performed with ease as it follows the first in first out rule.
- Queues are useful when a particular service is used by multiple consumers.
- Queues are fast in speed for data inter-process communication.
- Queues can be used in the implementation of other data structures.

Disadvantages of Queue:

- The operations such as insertion and deletion of elements from the middle are time consuming.
- Limited Space.
- In a classical queue, a new element can only be inserted when the existing elements are deleted from the queue.
- Searching an element takes O(N) time.
- Maximum size of a queue must be defined prior.

Pages Numbers from data structure and algorithms using python

Page number
1-6
33-47
69-75
155-165
169-175
179-180
193-206
218
221-237