1.

Stacks are a fundamental data structure(Last-In-First-Out (LIFO) principle),
where the last element added to the stack is the first one to be removed.
Real examples:

1. Undo/Redo Operations Many text editors(like word, notepad++), photo editing software(like photoshop), and other softwares that have undo/redo. These kind of features are using a stack data structure, when user perform an action(like typing text) , the action will be added to undo the undo stack, if user wants to undo an action, the top operation
   on the stack is popped and undone. The redo uses another stack to contain the undo actions, if the user wants to make a redo action, it is popped from the stack and redo.
2. Browser History Browsers use stack data structure to track the visited webs, when the user goes to a new web page, it is added to the stack, and when user click back button, the top web page is popped out of the stack, showing the previous visited page. The forward function works in similar way.
3. Supermarkets' trolleys The way how supermarkets organize trolleys, using a stack data structure, when a user give back a trolley, it is added at the last of the line, it is popped out of the line if another user is taking it out, following Last In First Out principle.

2.

Both sequential search and binary search are two ways used to search for an element in an array or a list or others. Sequential Search:

- Sequential search, also called linear search, is a searching algorithm that
- sequentially checks each element in the data structure until the target element is found or reaching the end of the data structure.(GeeksforGeeks, 2019)

Advantages:

- It is simple to implement and understand.
- It works on both sorted and unsorted data structures.
- It is efficient for small data or if the target element is closer to the beginning of the data set.

Disadvantages:

- Time complexity is O(n)(worst case), so it is inefficient for large data sets.
- All elements must be checked, even if there is no target element in the data set.

Binary Search: Comparing to sequential Search, binary search is a more efficient that works on a sorted data structure. (e.g. sorted array) (JavaPoint, 2011)

Advantages:

- Time complexity is O(log n) in the average and best cases, so it's much more efficient than a sequential search for large data sets.
- Only Half of the elements need to be re-search after each comparison, leading to faster searches.
- Ideal for large, sorted data structures.

Disadvantages:

- The data structure must be sorted before the implement of Binary Search.
- It is not suitable for unsorted data, as it cannot find target efficiently.(GeeksforGeeks, 2024)
- It's not always the best choice for big arrays, hash tables may be more efficient.

Reference

Works Cited“Applications, Advantages and Disadvantages of Binary Search.” GeeksforGeeks, 4 Mar. 2024, www.geeksforgeeks.org/applications-advantages-and-disadvantages-of-binary-search/.“Binary Search - Javatpoint.” Www.javatpoint.com, 2011, www.javatpoint.com/binary-search.GeeksforGeeks. “Linear Search - GeeksforGeeks.” GeeksforGeeks, Feb. 2019, www.geeksforgeeks.org/linear-search/.