Bellevue University

Stacks and Queues

Timothy Jelinek

CS420-T301 Data Structures

6/21/2024

Stacks are linear data structures that allow for the addition and removal of elements at the top of the stack. The practice of this addition and removal at the top is called last in, first out. Advantages of stacks include simplicity, efficiency, last in, first out for better use of function calls and expression evaluation, and limited memory usage. Disadvantages of stacks include limited access, potential of overflow, stacks are not suitable for random access, and stacks have limited capacity. The elements can only be accessed from the top of the stack, so it is difficult to retrieve and edit the elements in the middle. Adding more and more elements can cause an overflow error and cause losses in data. Stacks aren’t ideal when working with applications that have elements that you would want to access in a specific order. Stacks can’t store an unlimited number of variables, which can be inconvenient. Applications of stacks are function calls, recursion, expression evaluation, syntax parsing, and for memory management. Stacks are great data structures that are used every day by professionals all over. After reading all about stacks, it has been expressed that stacks should be used when working with local variables and recursion, and when checking the validity of syntax. Stacks are also useful when allocating and managing memory in operating systems and in programming languages.

Queues are also linear data structures, but they follow the first in, first out principle. Queues use the operation of enqueue to add elements to the back of the queue and dequeue to remove elements from the front of the queue. Queues are often used when managing data flow and handling tasks within operating systems, network protocols, and in data processing systems. Some ways queues are used include task scheduling, resource allocating, batch processing, message buffering, event handling, traffic managing, within operating systems, within network protocols, within printer queues, within web servers, and within breadth-first search algorithms. There are also issues when using queues in applications, which include, queue overflow, queue underflow, blocking queues, priority inversion, synchronization issues, memory management fragmentation, and reduced performance.

Stacks and queues are very similar, but they have key differences to make them better to use in different situations. Stacks use the same end for the addition and removal of elements, where queues use different ends for removal and addition. Stacks use last in, first out, while queues use first in, first out. Stacks use a pointer at the top which is used to track the element that was added most recently, while queues use two pointers, one at the front and one at the rear, to track the oldest added items and the most recently added elements. Stacks use push and pop operations, where queues use enqueue and dequeue. Stacks are full when their top is equal to max-1, and queues are max when the rear equals max-1. There are no variants of stacks, but there are three variants of queues, which are circular, queue, and priority queue. Finally, stacks can be shown vertically, and queues can be shown horizontally.

Sources:

GeeksforGeeks. (2024, March 5). *Applications, Advantages and Disadvantages of Stack*. GeeksforGeeks. https://www.geeksforgeeks.org/applications-advantages-and-disadvantages-of-stack/

GeeksforGeeks. (2023, March 28). *Applications of queue data structure*. GeeksforGeeks. https://www.geeksforgeeks.org/applications-of-queue-data-structure/

Testbook. (2023, August 30). *Difference between stack and queue data structures | Testbook.com*. Testbook. https://testbook.com/gate/difference-stack-and-queue-data-structures