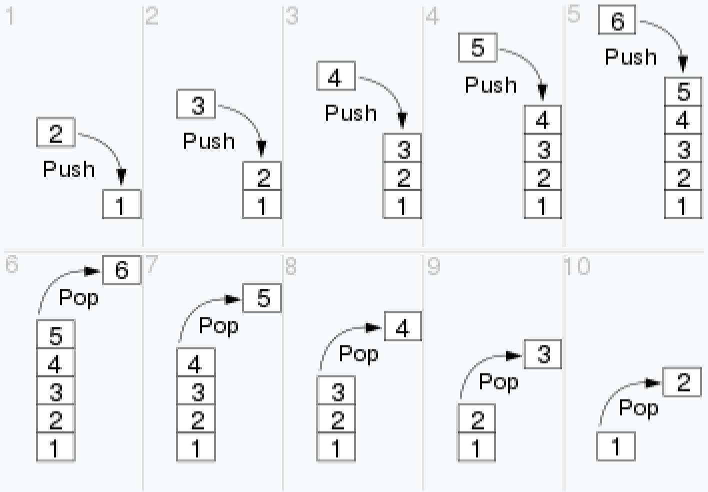
Stack (abstract data type)

For the use of the term LIFO in accounting, see LIFO (accounting). For the use of the term pushdown in strength training, see Pushdown (exercise).

For other uses, see Stack (disambiguation).

Similar to a stack of plates, adding or removing is only possible at the top.



Simple representation of a stack runtime with *push* and *pop* operations.

In computer science, a **stack** is an abstract data type that serves as a collection of elements, with two main operations:

- **Push**, which adds an element to the collection, and
- **Pop**, which removes the most recently added element that was not yet removed.

Additionally, a peek operation can, without modifying the stack, return the value of the last element added. Calling this structure a *stack* is by analogy to a set of physical items stacked one atop another, such as a stack of plates.

The order in which an element added to or removed from a stack is described as **last in, first out**, referred to by the acronym **LIFO**.[nb 1] As with a stack of physical objects, this structure makes it easy to take an item off the top of the stack, but accessing a datum deeper in the stack may require taking off multiple other items first.[1]

Considered as a linear data structure, or more abstractly a sequential collection, the push and pop operations occur only at one end of the structure, referred to as the *top* of the stack. This data structure makes it possible to implement a stack as a singly linked list and as a pointer to the top element. A stack may be implemented to have a bounded capacity. If the stack is full and does not contain enough space to accept another element, the stack is in a state of stack overflow.

A stack is needed to implement depth-first search.

A Stack is a linear data structure that follows the **LIFO (Last-In-First-Out)** principle. Stack has one end, whereas the Queue has two ends (**front and rear**). It contains only one pointer **top pointer** pointing to the topmost element of the stack. Whenever an element is added in the stack, it is added on the top of the stack, and the element can be deleted only from the stack. In other words, a ***stack can be defined as a container in which insertion and deletion can be done from the one end known as the top of the stack.***

Some key points related to stack

- It is called as stack because it behaves like a real-world stack, piles of books, etc.
- A Stack is an abstract data type with a pre-defined capacity, which means that it can store the elements of a limited size.
- It is a data structure that follows some order to insert and delete the elements, and that order can be LIFO or FILO.

Working of Stack

Stack works on the LIFO pattern. As we can observe in the below figure there are five memory blocks in the stack; therefore, the size of the stack is 5.

Suppose we want to store the elements in a stack and let's assume that stack is empty. We have taken the stack of size 5 as shown below in which we are pushing the elements one by one until the stack becomes full.

Since our stack is full as the size of the stack is 5. In the above cases, we can observe that it goes from the top to the bottom when we were entering the new element in the stack. The stack gets filled up from the bottom to the top.

When we perform the delete operation on the stack, there is only one way for entry and exit as the other end is closed. It follows the LIFO pattern, which means that the value entered first will be removed last. In the above case, the value 5 is entered first, so it will be removed only after the deletion of all the other elements.