Data Structures & Algorithms

Chapter - 05

Stack

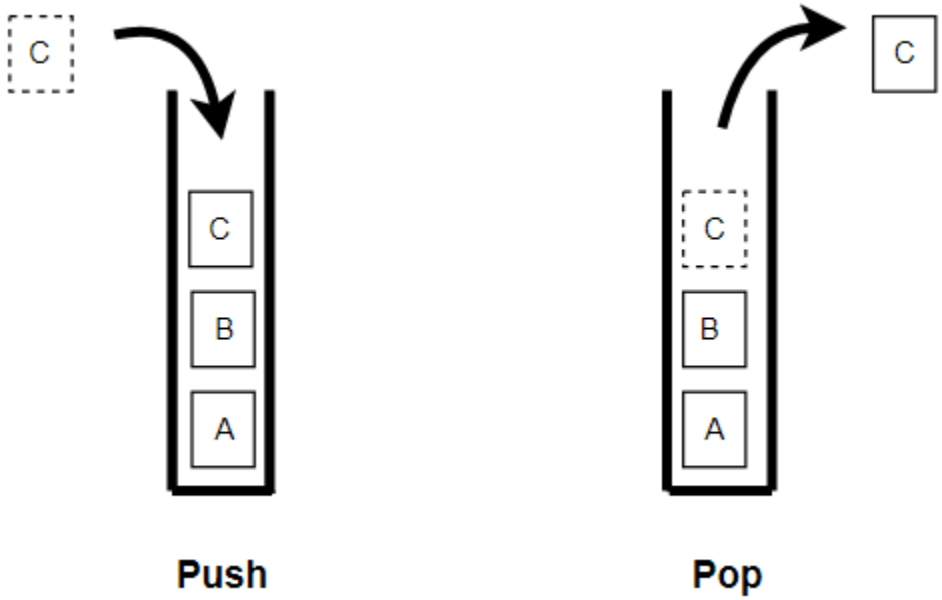
Rakin Mohammad Sifullah
Computer Science Engineer

Stack

A stack is a linear data structure that follows a particular order in which the operations are performed. The order may be Last In First Out (LIFO) or First In Last Out (FILO).
It has two main functions push and pop. Insertion in a stack is done using the push function and removal from a stack using the pop function.
The stack allows access to only the last element inserted hence, an item can be inserted or removed from the stack from one end called the top of the stack. It is, therefore, also called Last-In-First-Out (LIFO) list.

Stack has three properties:

1. capacity stands for the maximum number of elements the stack can hold,
2. Size stands for the current size of the stack and
3. Elements are an array of elements.



There are many real-life examples of a stack. Consider an example of plates stacked over one another in the canteen. The plate which is at the top is the first one to be removed, i.e. the plate which has been placed at the bottommost position remains in the stack for the longest period of time. So, it can be simply seen to follow LIFO(Last In First Out)/FILO(First In Last Out) order.

Functions/Basic Operations on Stack:

Push: Adds an item to the stack. If the stack is full, then it is said to be an Overflow condition.

```
begin
 if stack is full
    return
 endif
else
 increment top
 stack[top] assign value
end else
end procedure
```

Pop: Removes an item from the stack. The items are popped in the reversed order in which they are pushed. If the stack is empty, then it is said to be an Underflow condition.

```
begin
 if stack is empty
    return
 endif
else
 store value of stack[top]
 decrement top
 return value
end else
end procedure
```

Top: Returns the top element of the stack.

```
begin
  return stack[top]
end procedure
```

isEmpty: Returns true if the stack is empty, else false.

```
begin
 if top < 1
    return true
 else
    return false
end procedure
```

Property:

1. Each function runs in O(1) time.
2. It has two basic implementations
    a. Array-based implementation – It is simple and efficient but the maximum size of the stack is fixed.
    b. Singly Linked List-based implementation – It is complicated but there is no limit on the stack size, it is subjected to the available memory.