CHANDIGARH UNIVERSITY
Discover. Learn. Empower.

Stack
(CAT-201)

Design By:
Ms. Gurpreet kaur dhiman
Ms.Mandeep kaur
Chandigarh University-Gharuan

STACK

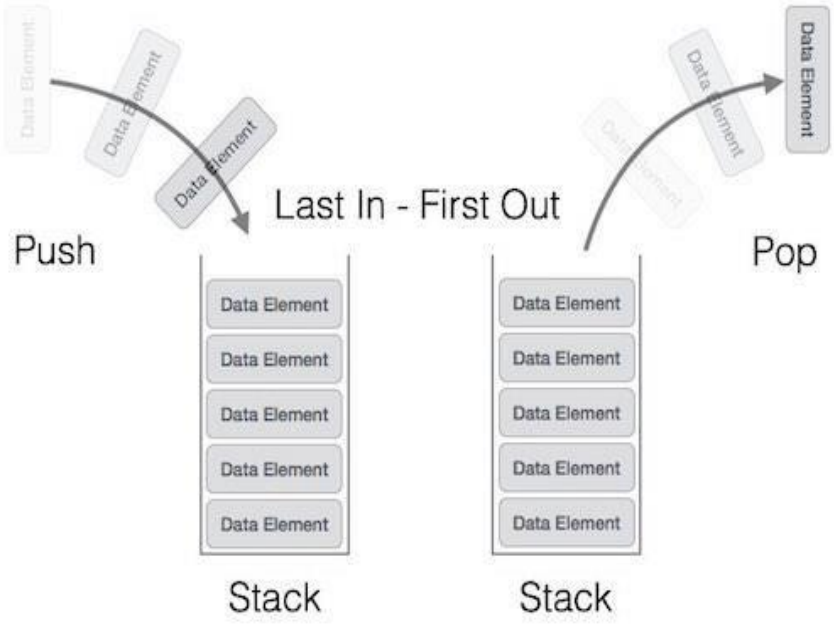
CU
CHANDIGARH UNIVERSITY

- A stack is an Abstract Data Type (ADT), commonly used in most programming languages. It is named stack as it behaves like a real-world stack, for example – a deck of cards or a pile of plates, etc.

- A real-world stack allows operations at one end only. For example, we can place or remove a card or plate from the top of the stack only. Likewise, Stack ADT allows all data operations at one end only. At any given time, we can only access the top element of a stack.

STACK

A stack can be implemented by means of Array, Structure, Pointer, and Linked List. Stack can either be a fixed size one or it may have a sense of dynamic resizing. Here, we are going to implement stack using arrays, which makes it a fixed size stack implementation.



Reference:
https://www.tutorialspoint.com/data_structures_algorithms/stack_algorithm.htm

STACK OPERATION

CU
CHANDIGARH UNIVERSITY

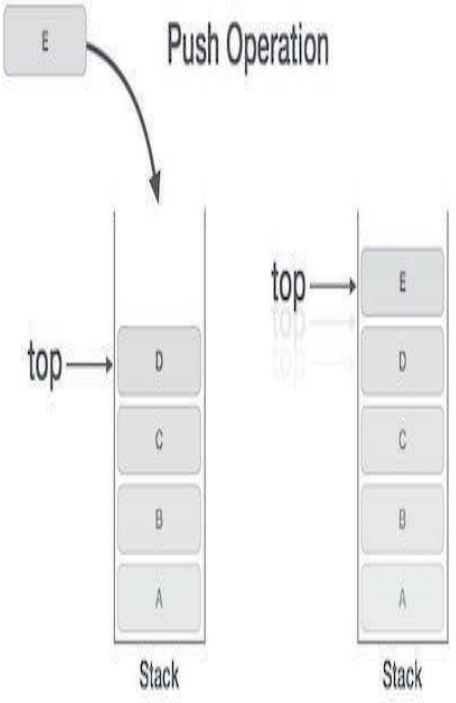
Stack operations may involve initializing the stack, using it and then de-initializing it. Apart from these basic stuffs, a stack is used for the following two primary operations −

- **push()** − Pushing (storing) an element on the stack.
- **pop()** − Removing (accessing) an element from the stack.
- When data is Pushed onto stack.
- To use a stack efficiently, we need to check the status of stack as well. For the same purpose, the following functionality is added to stacks −
- **peek()** − get the top data element of the stack, without removing it.
- **isFull()** − check if stack is full.
- **isEmpty()** − check if stack is empty.

STACK(INSERTION)

Push Operation

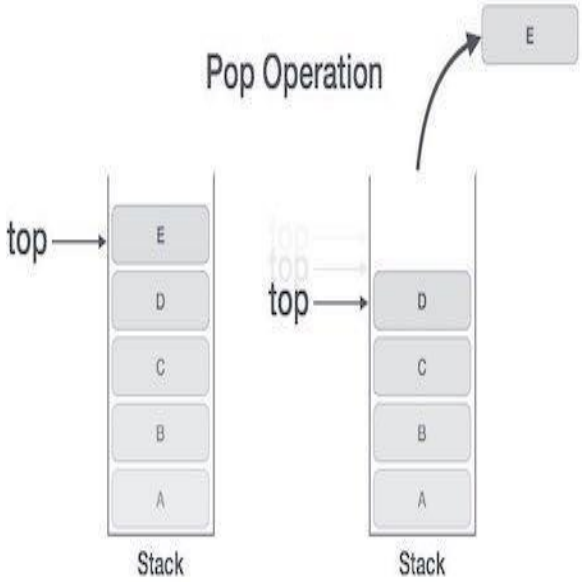
- The process of putting a new data element onto stack is known as a Push Operation. Push operation involves a series of steps −
- **Step 1** − Checks if the stack is full.
- **Step 2** − If the stack is full, produces an error and exit.
- **Step 3** − If the stack is not full, increments **top** to point next empty space.
- **Step 4** − Adds data element to the stack location, where top is
- pointing.
- **Step 5** − Returns success.



https://www.tutorialspoint.com/data_structures_algorithms/stack_algorithm.htm

STACK(DELETION)

- Accessing the content while removing it from the stack, is known as a Pop Operation. In an array implementation of pop() operation, the data element is not actually removed, instead **top** is decremented to a lower position in the stack to point to the next value.
- A Pop operation may involve the following steps −
- **Step 1** − Checks if the stack is empty.
- **Step 2** − If the stack is empty, produces an error and exit.
- **Step 3** − If the stack is not empty, accesses the data element
- at which **top** is pointing.
- **Step 4** − Decreases the value of top by 1.
- **Step 5** − Returns success.



https://www.tutorialspoint.com/data_structures_algorithms/stack_algorithm.htm

STACK PUSH PROCDURE

begin procedure push: stack, data

if stack is full

return null

endif

top ← top + 1

stack[top] ← data

end procedure

STACK POP PROCEDURE

begin procedure POP: stack, data

if stack is EMPTY

return null

endif

DATA ← stack[top]

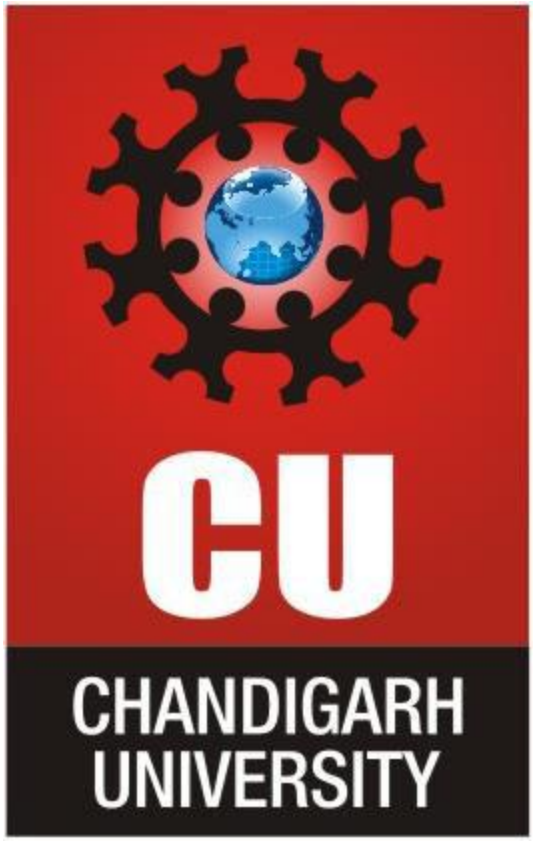
TOP← TOP-1

RETURN DATA

end procedure

Bibliography

- Seymour Lipschutz, Schaum's Outlines Series Data structures TMH
- Introduction to Data Structures Applications, Trembley&Soreson, Second Edition, Pearson Education
- www.geeksforgeeks.org/analysis-of-algorithms-set-3asymptotic-notations/
- www.tutorialspoint.com/data_structures_algorithms/asymptotic_analysis.htm



Thank You