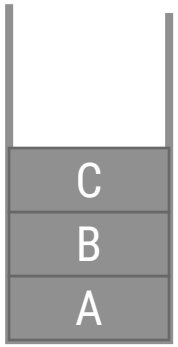
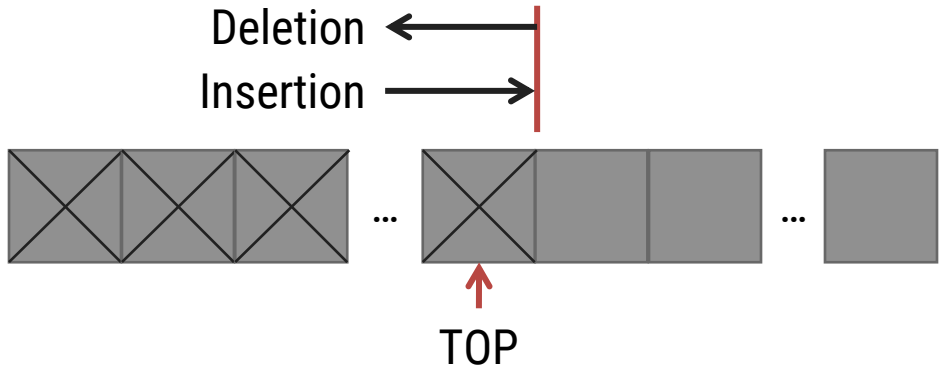
Stack

- A linear list which allows insertion and deletion of an element at one end only is called ***stack***.
- The insertion operation is called as ***PUSH*** and deletion operation as ***POP***.
- The most accessible elements in stack is known as ***top***.
- The elements can only be removed in the opposite orders from that in which they were added to the stack.
- Such a linear list is referred to as a ***LIFO (Last In First Out)*** list.





Stack Cont…

- A pointer TOP keeps track of the top element in the stack.
- Initially, when the ***stack is empty***, **TOP** has a value of ***"zero"***.
- Each time a **new element is inserted** in the stack, the pointer is ***incremented by "one"*** before, the element is placed on the stack.
- The pointer is ***decremented by "one"*** each time a **deletion** is made from the stack.

Applications of Stack

- Recursion
- Keeping track of function calls
- Evaluation of expressions
- Reversing characters
- Servicing hardware interrupts
- Solving combinatorial problems using backtracking
- Expression Conversion (Infix to Postfix, Infix to Prefix)
- Game Playing (Chess)
- Microsoft Word (Undo / Redo)
- Compiler – Parsing syntax & expression
- Finding paths

Procedure : PUSH (S, TOP, X)

- This procedure inserts an element **X** to the top of a stack.
- Stack is represented by a vector **S** containing **N** elements.
- A pointer **TOP** represents the top element in the stack.

```
1. [Check for stack overflow]
      If    TOP ≥ N
      Then  write ('STACK OVERFLOW')
            Return
2. [Increment TOP]
      TOP ← TOP + 1
3. [Insert Element]
      S[TOP] ← X
4. [Finished]
      Return
```

Stack is empty, TOP = 0, N=3

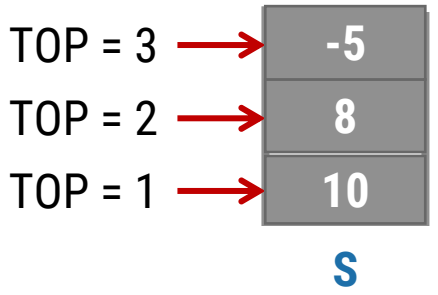
PUSH(S, TOP, 10)

PUSH(S, TOP, 8)

PUSH(S, TOP, -5)

PUSH(S, TOP, 6)

Overflow



Function : POP (S, TOP)

- This function ***removes & returns*** the top element from a stack.
- Stack is represented by a vector **S** containing **N** elements.
- A pointer **TOP** represents the top element in the stack.

```
1. [Check for stack underflow]
      If    TOP = 0
      Then  write ('STACK UNDERFLOW')
            Return (0)
2. [Decrement TOP]
      TOP ← TOP - 1
3. [Return former top element of
   stack]
      Return(S[TOP + 1])
```

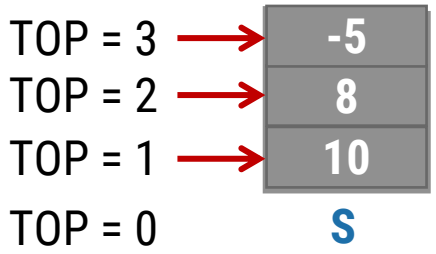
POP(S, TOP)

POP(S, TOP)

POP(S, TOP)

POP(S, TOP)

Underflow



Function : PEEP (S, TOP, I)

- This function returns the value of the **Ith** element from the **TOP** of the stack. The element is not deleted by this function.
- Stack is represented by a vector **S** containing **N** elements.

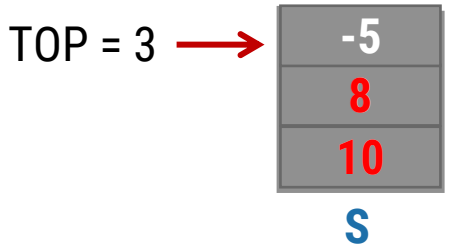
```
1. [Check for stack underflow]
      If    TOP-I+1 ≤ 0
      Then  write ('STACK UNDERFLOW')
            Return (0)
2. [Return Ith element from top
    of the stack]
      Return(S[TOP-I+1])
```

PEEP (S, TOP, 2)

PEEP (S, TOP, 3)

PEEP (S, TOP, 4)

Underflow



PROCEDURE : CHANGE (S, TOP, X, I)

- This procedure changes the value of the **Ith** element from the top of the stack to **X**.
- Stack is represented by a vector **S** containing **N** elements.

```
1. [Check for stack underflow]
      If    TOP-I+1 ≤ 0
      Then  write ('STACK UNDERFLOW')
            Return
2. [Change I^th element from top
    of the stack]
      S[TOP-I+1] ← X
3. [Finished]
      Return
```

CHANGE (S, TOP, 50, 2)

CHANGE (S, TOP, 9, 3)

CHANGE (S, TOP, 25, 8)

Underflow