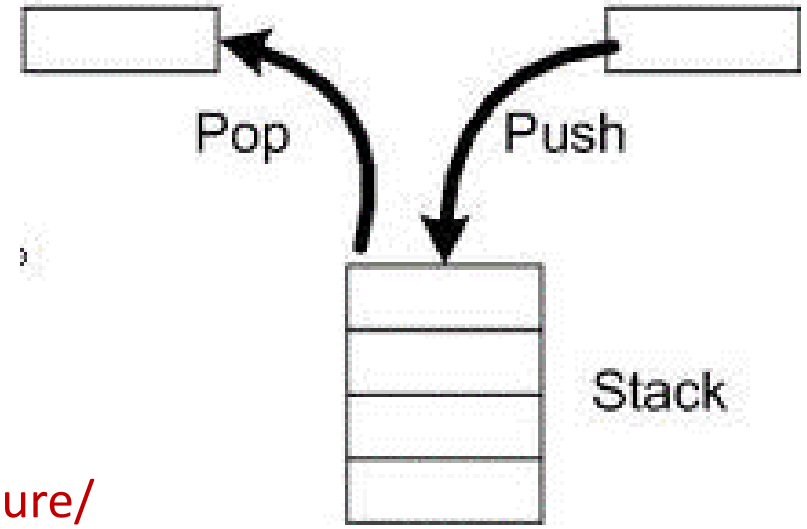
PART 2
About **Stack**

Stack

Stack is a linear data structure

0rder/concept : LIFO(Last In First Out) or FILO(First In Last Out).



Read: https://www.geeksforgeeks.org/stack-data-structure/

Stack Implementation

Implement Stack : Use *Java Collection framework* that provides a *Stack class*. (by import **java.util.Stack** class). It is a package that contains the collection of stack classes. It represents a last-in-first-out (LIFO) stack of objects

Common Methods of a stack:

- `Push` – to add new element into a stack
- Pop – to remove the top element from a stack

Other stack Methods:

- `Peek` - to retrieve the element present at the top of the stack without removing it.
- `Empty`- to check whether a stack is empty or not.
- `Size` - used to get the size of the Stack or the number of elements present in the Stack

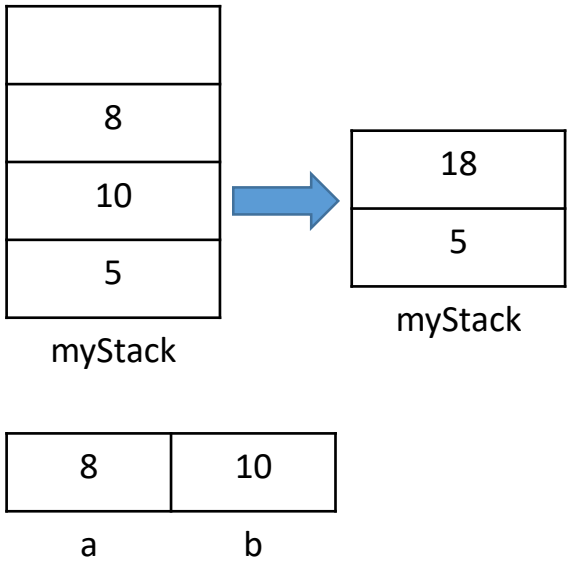
Example:

```
import java.util.Stack;

Stack<String> STACK = new Stack<String>(); // to create a stack Object of type String
String a;
```

Syntax	Code Example	Objective
stack.push (*E element*);	STACK.push(“5”); STACK.push(“Welcome”);	to push an element into the Stack Push method accepts one parameter *element* of type Stack to be pushed into the stack.
STACK.pop();	`a=STACK.pop();` `System.out.println ("Remove : " + STACK.pop());`	Remove the top element of stack pop method does not take any parameters.
STACK.peek();	`System.out.println ("Top element: " + STACK.peek());`	to retrieve the top element without removing it. peek method returns the top element or returns NULL if the Stack is empty.
STACK.empty();	`if !(STACK.empty()` `System.out.println (“Remove : " +STACK.pop());`	Empty method does not take any parameters. It returns Boolean value (true if the stack is empty else return false)
STACK.size();	`System.out.println (“Number of elements in Stack: " + myStack.size());`	get the size of the Stack

Example of Stack program

```
import java.util.Stack;
import java.util.*;
public class StackDemo {
  public static void main(String args[])
  {
      Stack<Integer> myStack = new Stack<Integer>(); // Create myStack object

      myStack.push(5);
      myStack.push(10);
      myStack.push(8);
      a=myStack.pop();
      b=myStack.pop();
      myStack.push(a+b);
      System.out.println (“Top element: " + myStack.peek());
      System.out.println (“Number of elements in myStack: " + myStack.size());
      System.out.println (“Final myStack: " + myStack);
   }
}
```



Demonstration

Refer to links https://www.geeksforgeeks.org/stack-data-structure/