STACK

```
package constructorpac;

import java.util.Collections;
import java.util.Stack;

//Stack is a linear data Structure / generic data structure
//it follows the Last In First Out (LIFO) principle
//maintain insertion order
//allow duplicate and null values
//Dynamic in size
//implements list ,Extends Vector, random access allowed.


public class StackImple {

	public static void main(String[] args) {
		Stack <Integer> sk=new Stack <>();  // Creating a new stack

//push elements onto the stack instead of adding using push() method

		sk.push(1);
		sk.push(2);
		sk.push(3);
		sk.push(4);

		System.out.println(sk);

// pop elements from the stack instead of removing using pop() method

		System.out.println("The removed/poped element is : "+sk.pop());
		System.out.println(sk);

// accessing the element which is top in the stack using peek() method

		System.out.println("The top element is : "+sk.peek());

		System.out.println(sk.reversed()); //it reversed but not actually reversed in the memory
		System.out.println(sk);


	//________________________________________________________________________________
		// element can be added using add() and inserted between the stack becoz it extends
vector
```

```
                // but it violets the stack protocol
                // it deviates the standards of stack
                // it is not recommended becoz push is designed for (LIFO) stack
                //and insertion is not allowed in principle of stack

                sk.add(5);
                sk.insertElementAt(0, 0);
                sk.insertElementAt(4, 4);
                System.out.println(sk);

        //______________________________________________________________________

                Stack <String> sk2=new Stack <>();

                sk2.push("Vivo");
                sk2.push("Oppo");
                sk2.push("Samsung");
                sk2.push("Apple");

                System.out.println(sk2);

// searching an element in  stack
//if not found it returns -1

                int pos=sk2.search("Apple");
                if(pos==-1) {
                        System.out.println("The element not found in the stack");
                }
                else {
                        System.out.println("The element found in the position "+pos);
                }

//sorting the stack

                Collections.sort(sk2);
                System.out.println(sk2);
                System.out.println("The element after sort found at the position"+sk2.search("Apple"));

//Checking the stack is empty or not

                if(sk.isEmpty()) {
                        System.out.println("The stack is empty");
                }
                else {
                        System.out.println("The stack contains elements");
                }

// default method of creating stack
```

```
                    Stack sk3 =new Stack();

                    sk3.push(4); // but it's not type safe we use only generic stack
                    sk3.push("Game");
                    sk3.push(5.8);
                    System.out.println(sk3);



          }

}
```