**Your task is to implement  2 stacks in one array efficiently. You need to implement 4 methods.**

**twoStacks : Initialize the data structures and variables to be used to implement  2 stacks in one array.**

**push1 : pushes element into first stack.**

**push2 : pushes element into second stack.**

**pop1 : pops element from first stack and returns the popped element. If first stack is empty, it should return -1.**

**pop2 : pops element from second stack and returns the popped element. If second stack is empty, it should return -1.**

**Example 1:**

**Input:**
**push1(2)**
**push1(3)**
**push2(4)**
**pop1()**
**pop2()**
**pop2()**
**Output:**
**3 4 -1**
**Explanation:**
**push1(2) the stack1 will be {2}**
**push1(3) the stack1 will be {2,3}**
**push2(4) the stack2 will be {4}**
**pop1()  the poped element will be 3 from stack1 and stack1 will be {2}**
**pop2()  the poped element will be 4 from stack2 and now stack2 is empty**
**pop2()  the stack2 is now empty hence returned -1.**

**Example 2:**

**Input:**
**push1(1)**
**push2(2)**
**pop1()**
**push1(3)**
**pop1()**
**pop1()**
**Output:**
**1 3 -1**
**Explanation:**

**push1(1) the stack1 will be {1}**
**push2(2) the stack2 will be {2}**

**pop1()  the poped element will be 1 from stack1 and stack1 will be empty**

**push1(3) the stack1 will be {3}**
**pop1()  the poped element will be 3 from stack1 and stack1 will be empty**

**pop1()  the stack1 is now empty hence returned -1.**

**Your Task:**

**You don't need to read input or print anything. You are required to complete the 4 methods push1, push2 which takes one argument an integer 'x' to be pushed into stack one and two and pop1, pop2 which returns the integer poped out from stack one and two. If no integer is present in the stack return -1.**

**Expected Time Complexity: O(1) for all the four methods.**

**Expected Auxiliary Space: O(1) for all the four methods.**

**Constraints:**

**1 <= Number of queries <= 104**

**1 <= Number of elements in the stack <= 100**

**The sum of count of elements in both the stacks < size of the given array**

## Solution:

```
class twoStacks {
    Stack<Integer> st1;
    Stack<Integer> st2;
    twoStacks() {
        st1=new Stack<Integer>();
        st2=new Stack<Integer>();
    }

    // Function to push an integer into the stack1.
    void push1(int x) {
        st1.push(x);
    }

    // Function to push an integer into the stack2.
    void push2(int x) {
        st2.push(x);
    }

    // Function to remove an element from top of the stack1.
    int pop1() {
        if(!st1.isEmpty())  return st1.pop();
        return -1;
    }

    // Function to remove an element from top of the stack2.
    int pop2() {
        if(!st2.isEmpty())  return st2.pop();
        return -1;
    }
}
```