/\* Implement a Stack using an array.

• Test Case 1:

Input: Push 5, 3, 7, Pop

Output: Stack = [5, 3], Popped element = 7

• Test Case 2:

Input: Push 10, Push 20, Pop, Push 15

Output: Stack = [10, 15], Popped element = 20 \*/

class Stack {

private int arr[];

private int top;

private int capacity;

Stack(int size) {

arr = new int[size];

capacity = size;

top = -1;

}

public void push(int x) {

if (top == capacity - 1) {

System.out.println("Overflow");

System.exit(1);

}

arr[++top] = x;

}

public int pop() {

if (top == -1) {

System.out.println("Underflow");

System.exit(1);

}

return arr[top--];

}

public void printStack() {

System.out.print("Stack = [");

for (int i = 0; i <= top; i++) {

System.out.print(arr[i]);

if (i < top) System.out.print(", ");

}

System.out.println("]");

}

}

public class question1 {

public static void main(String[] args) {

// Test Case 1

Stack stack1 = new Stack(5);

stack1.push(5);

stack1.push(3);

stack1.push(7);

int poppedElement1 = stack1.pop();

stack1.printStack();

System.out.println("Popped element = " + poppedElement1);

// Test Case 2

Stack stack2 = new Stack(5);

stack2.push(10);

stack2.push(20);

int poppedElement2 = stack2.pop();

stack2.push(15);

stack2.printStack();

System.out.println("Popped element = " + poppedElement2);

}

}

FLOWCHART  
 +---------+

| Start |

+---------+

|

v

+-------------------+

| Initialize Stack |

+-------------------+

|

v

+-------------------+

| Perform Operations|

| Push/Pop |

+-------------------+

|

v

+-------------------+

| Print Output |

+-------------------+

|

v

+---------+

| End |

+---------+  
  
Time complexity and space complexity  
time -0(1)  
Space-0(n)  
  
  
EXPLANATION  
**Creating a Stack:**

Stack stack1 = new Stack(5); initializes a stack with a capacity of 5 elements.

**Test Case 1 Execution:**

**Push 5**: Adds 5 to the stack. Stack becomes [5].

**Push 3**: Adds 3 to the stack. Stack becomes [5, 3].

**Push 7**: Adds 7 to the stack. Stack becomes [5, 3, 7].

**Pop**: Removes the top element (7). Stack becomes [5, 3].

Prints the stack and popped element: Stack = [5, 3], Popped element = 7.

**Test Case 2 Execution:**

**Push 10**: Adds 10 to the stack. Stack becomes [10].

**Push 20**: Adds 20 to the stack. Stack becomes [10, 20].

**Pop**: Removes the top element (20). Stack becomes [10].

**Push 15**: Adds 15 to the stack. Stack becomes [10, 15].

Prints the stack and popped element: Stack = [10, 15], Popped element = 20.

OUTPUT  
