1. **ArrayStack Class**

**public class ArrayStack implements Stack {**

**private int size;**

**private Object array[];**

**public ArrayStack(int capacity) {**

**array = new Object[capacity];**

**}**

**@Override**

**public Object peek() {**

**if (this.isEmpty()) {**

**throw new IllegalStateException("Stack is empty!");**

**}**

**return array[size - 1];**

**}**

**@Override**

**public Object pop() {**

**if (this.isEmpty()) {**

**throw new IllegalStateException("Stack is empty!");**

**}**

**Object obj = array[--size];**

**array[size] = null;**

**return obj;**

**}**

**@Override**

**public void push(Object obj) {**

**if (size == array.length) {**

**resizeArray();**

**}**

**array[size++] = obj;**

**}**

**@Override**

**public int size() {**

**return size;**

**}**

**@Override**

**public boolean isEmpty() {**

**return (size == 0);**

**}**

**public void resizeArray() {**

**Object newArray[] = this.array;**

**this.array = new Object[2 \* size];**

**System.arraycopy(newArray, 0, this.array, 0, newArray.length);**

**}**

**/////toString() converts all data of current object into an string**

**public String toString() {**

**if (this.isEmpty()) {**

**return "";**

**}**

**String string = "[";**

**for (int i = size - 1; i >= 0; i--) { //as it is an stack so last element willl be the first element**

**string += String.valueOf(array[i] + ",");**

**}**

**string = string.substring(0, string.lastIndexOf(',')) + "]";**

**return string;**

**}**

**/////equals() compares two stacks**

**public boolean equals(Stack obj) {**

**if (this.size() != obj.size()) {**

**return false;**

**}**

**Object array[] = new Object[this.size()];**

**Object array2[] = new Object[this.size()]; //as size of both are equal**

**boolean areEqual = true;**

**int i = 0; //counter variable**

**for (; i < array.length; i++) {**

**array[i] = this.pop(); //storing elements in array by popping so that we can store later same elements**

**array2[i] = obj.pop();**

**if (!array[i].equals(array2[i])) {**

**areEqual = false;**

**}**

**}**

**while (--i >= 0) {**

**this.push(array[i]); //again inserting those elements in**

**obj.push(array2[i]);**

**}**

**return areEqual;**

**}**

**////findLast() finds last element in the stack**

**public Object findLast() {**

**if (this.isEmpty()) {**

**throw new IllegalStateException("Stack is empty!");**

**}**

**return array[0];**

**}**

**/////toLinkedStack() returns LinkedStack object equivalent to curent ArrayStack object**

**public LinkedStack toLinkedStack() {**

**if (this.isEmpty()) {**

**return null;**

**}**

**LinkedStack stack = new LinkedStack();**

**for (int i = 0; i < this.size; i++) {**

**stack.push(array[i]);**

**}**

**return stack;**

**}**

**public static void main(String[] args) {**

**ArrayStack stack = new ArrayStack(2);**

**stack.push(30);**

**stack.push("Hello");**

**stack.push(20);**

**ArrayStack stack2 = new ArrayStack(2);**

**stack2.push(30);**

**stack2.push("Hello");**

**stack2.push(21);**

**System.out.println("stack.toString(): " + stack.toString());**

**System.out.println("stack2.toString(): "+stack2.toString());**

**System.out.println("stack.equals(stack2): " + stack.equals(stack2));**

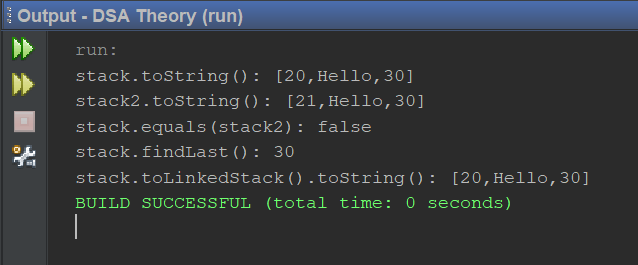
**System.out.println("stack.findLast(): " + stack.findLast());**

**System.out.println("stack.toLinkedStack().toString(): " + stack.toLinkedStack().toString());**

**}**

**}**

**OUTPUT**



1. **LinkedStack Class**

**import java.util.\*;**

**public class LinkedStack implements Stack {**

**private int size;**

**private Node top;**

**private class Node {**

**private Object object;**

**private Node next;**

**public Node(Object object, Node next) {**

**this.object = object;**

**this.next = next;**

**}**

**}**

**@Override**

**public Object peek() {**

**if (this.isEmpty()) {**

**throw new NoSuchElementException("Stack is Empty!");**

**}**

**return top.object;**

**}**

**@Override**

**public Object pop() {**

**if (this.isEmpty()) {**

**throw new NoSuchElementException("Stack is Empty!");**

**}**

**Object obj = top.object;**

**top = top.next;**

**--size;**

**return obj;**

**}**

**@Override**

**public void push(Object obj) {**

**top = new Node(obj, top);**

**size++;**

**}**

**@Override**

**public int size() {**

**return size;**

**}**

**@Override**

**public boolean isEmpty() {**

**return (size == 0);**

**}**

**/////toString() converts all data of current object into an string**

**public String toString() {**

**if (this.isEmpty()) {**

**return "";**

**}**

**String string = "[";**

**for (Node i = top; i != null; i = i.next) {**

**string += String.valueOf(i.object + ",");**

**}**

**string = string.substring(0, string.lastIndexOf(',')) + "]";**

**return string;**

**}**

**/////equals() compares two stacks**

**public boolean equals(Stack obj) {**

**if (this.size() != obj.size()) {**

**return false;**

**}**

**Object array[] = new Object[this.size()];**

**Object array2[] = new Object[this.size()]; //as size of both are equal**

**boolean areEqual = true;**

**int i = 0; //counter variable**

**for (; i < array.length; i++) {**

**array[i] = this.pop(); //storing elements in array by popping so that we can store later same elements**

**array2[i] = obj.pop();**

**if (!array[i].equals(array2[i])) {**

**areEqual = false;**

**}**

**}**

**while (--i >= 0) {**

**this.push(array[i]); //again inserting those elements in**

**obj.push(array2[i]);**

**}**

**return areEqual;**

**}**

**////findLast() finds last element in the stack**

**public Object findLast() {**

**if (this.isEmpty()) {**

**throw new NoSuchElementException("Stack is empty!");**

**}**

**Node i = top;**

**for (; i.next != null; i = i.next) //because if i.next=null it means i will be pointing to the last Object**

**{**

**}**

**return i.object;**

**}**

**/////toArrayStack() returns ArrayStack object equivalent to curent LinkedStack object**

**public ArrayStack toArrayStack() {**

**if (this.isEmpty()) {**

**return null;**

**}**

**Object array[] = new Object[this.size];**

**int count = 0;**

**ArrayStack stack = new ArrayStack(this.size);**

**for (Node i = top; i != null; i = i.next) {**

**array[count++] = i.object; //for preserving same order that's why storing elements in an Object array**

**}**

**for (int i = this.size - 1; i >= 0; i--) {**

**stack.push(array[i]);**

**}**

**return stack;**

**}**

**public static void main(String[] args) {**

**LinkedStack stack = new LinkedStack();**

**stack.push(40);**

**stack.push("Hello");**

**stack.push(200);**

**ArrayStack stack2 = new ArrayStack(2);**

**stack2.push(40);**

**stack2.push("Hello");**

**stack2.push(200);**

**System.out.println("stack.toString(): " + stack.toString());**

**System.out.println("stack2.toString(): " + stack2.toString());**

**System.out.println("stack.equals(stack2): " + stack.equals(stack2));**

**System.out.println("stack.findLast(): " + stack.findLast());**

**System.out.println("stack.toArrayStack().toString(): " + stack.toArrayStack().toString());**

**}**

**}**

**OUTPUT**

