Stack

class Stack {

// store elements of stack

private int arr[];

// represent top of stack

private int top;

// total capacity of the stack

private int capacity;

// Creating a stack

Stack(int size) {

// initialize the array

// initialize the stack variables

arr = new int[size];

capacity = size;

top = -1;

}

// push elements to the top of stack

public void push(int x) {

if (isFull()) {

System.out.println("Stack OverFlow");

// terminates the program

System.exit(1);

}

// insert element on top of stack

System.out.println("Inserting " + x);

arr[++top] = x;

}

// pop elements from top of stack

public int pop() {

// if stack is empty

// no element to pop

if (isEmpty()) {

System.out.println("STACK EMPTY");

// terminates the program

System.exit(1);

}

// pop element from top of stack

return arr[top--];

}

// return size of the stack

public int getSize() {

return top + 1;

}

// check if the stack is empty

public Boolean isEmpty() {

return top == -1;

}

// check if the stack is full

public Boolean isFull() {

return top == capacity - 1;

}

// display elements of stack

public void printStack() {

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

System.out.print(arr[i] + ", ");

}

}

public static void main(String[] args) {

Stack stack = new Stack(5);

stack.push(1);

stack.push(2);

stack.push(3);

System.out.print("Stack: ");

stack.printStack();

// remove element from stack

stack.pop();

System.out.println("\nAfter popping out");

stack.printStack();

}

}