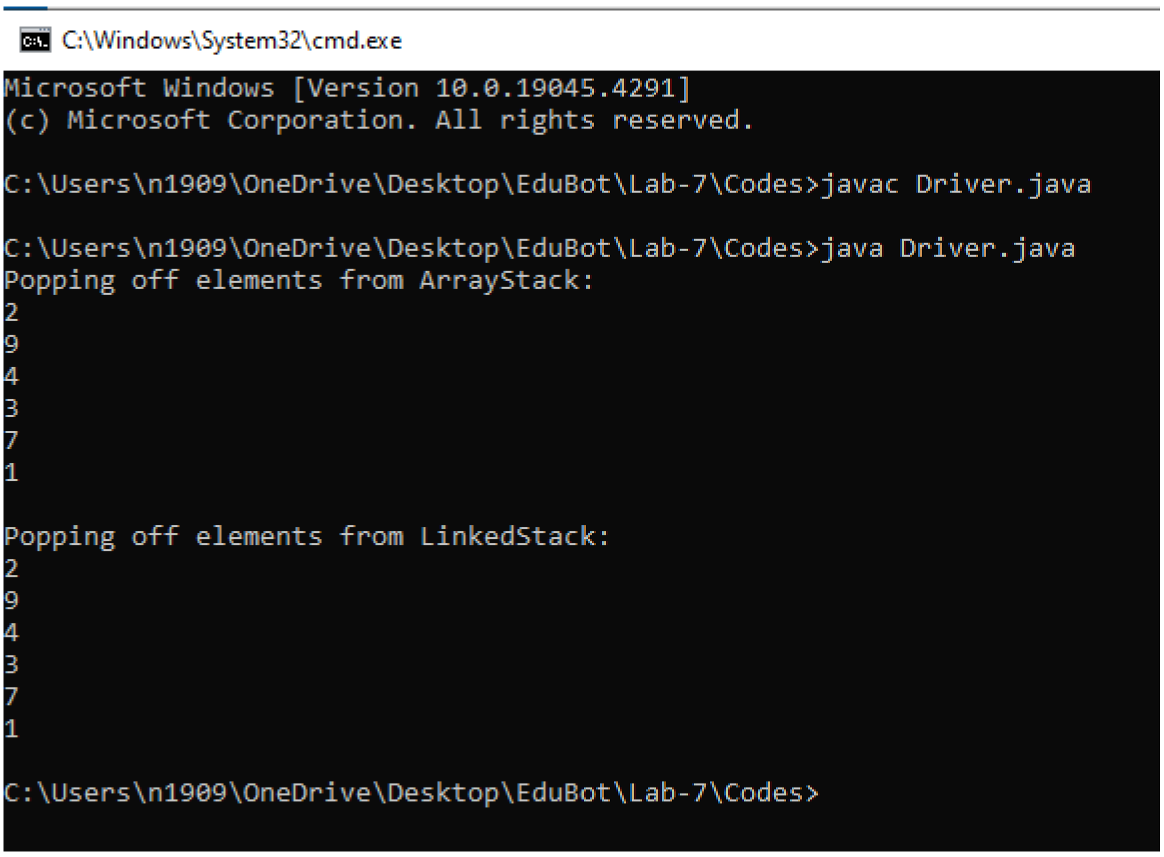
Programming Fundamentals
Lab #7

Exercise 1:

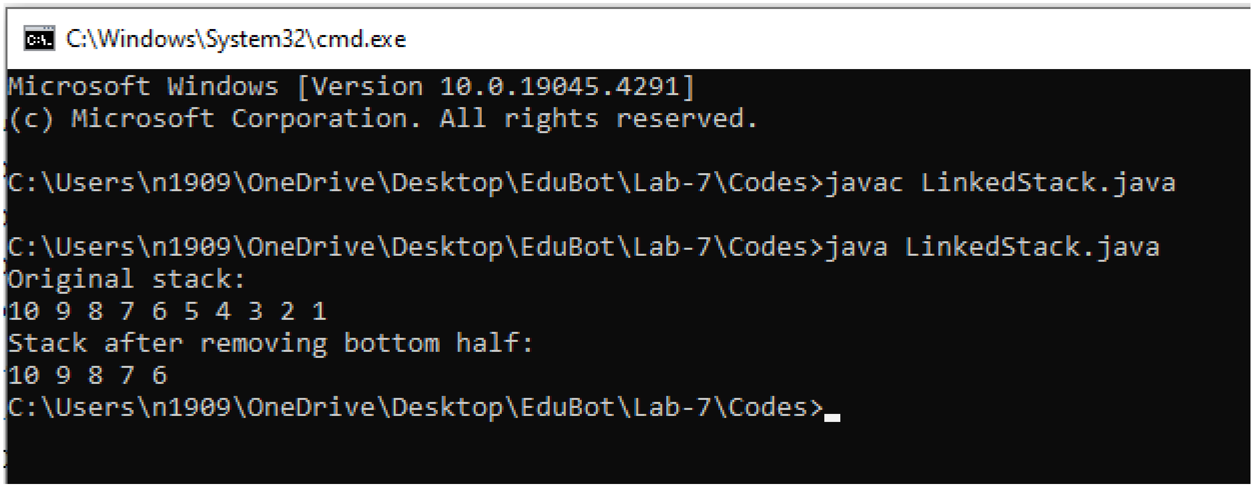
```
public class Driver {
    public static void main(String[] args) {
        ArrayStack arrayStack = new ArrayStack();
        LinkedStack linkedStack = new LinkedStack();

        int[] numbers = {1, 7, 3, 4, 9, 2};

        // Pushing numbers onto the stacks
        for (int num : numbers) {
            arrayStack.push(num);
            linkedStack.push(num);
        }

        System.out.println("Popping off elements from ArrayStack:");
        while (!arrayStack.isEmpty()) {
            System.out.println(arrayStack.pop());
        }

        System.out.println("\nPopping off elements from LinkedStack:");
        while (!linkedStack.isEmpty()) {
            System.out.println(linkedStack.pop());
        }
    }
}

class ArrayStack {
    private int[] array;
    private int top;

    public ArrayStack() {
        array = new int[10]; // Assuming initial capacity of 10
        top = -1;
    }

    public void push(int item) {
        if (top == array.length - 1) {
            // Resize array if full
            int[] newArray = new int[array.length * 2];
            System.arraycopy(array, 0, newArray, 0, array.length);
            array = newArray;
        }
        array[++top] = item;
    }
    public int pop() {
        if (isEmpty()) {
            throw new IllegalStateException("Stack is empty");
        }
        return array[top--];
    }

    public boolean isEmpty() {
        return top == -1;
    }
}

class LinkedStack {
    private Node top;

    private class Node {
        int data;
        Node next;

        Node(int data) {
            this.data = data;
        }
    }

    public void push(int item) {
        Node newNode = new Node(item);
        newNode.next = top;
        top = newNode;
    }

    public int pop() {
        if (isEmpty()) {
            throw new IllegalStateException("Stack is empty");
        }
        int data = top.data;
        top = top.next;
        return data;
    }

    public boolean isEmpty() {
        return top == null;
    }
}
```



```
Microsoft Windows [Version 10.0.19045.4291]
(c) Microsoft Corporation. All rights reserved.

C:\Users\n1909\OneDrive\Desktop\EduBot\Lab-7\Codes>javac Driver.java

C:\Users\n1909\OneDrive\Desktop\EduBot\Lab-7\Codes>java Driver.java
Popping off elements from ArrayStack:
2
9
4
3
7
1

Popping off elements from LinkedStack:
2
9
4
3
7
1

C:\Users\n1909\OneDrive\Desktop\EduBot\Lab-7\Codes>
```

Exercise 1:

```
import java.util.EmptyStackException;

public class LinkedStack<E> {

    private static class Node<E> {
        private E element;
        private Node<E> next;
        public Node(E e, Node<E> n) {
            element = e;
            next = n;
        }
        public E getElement() {
            return element;
        }
        public Node<E> getNext() {
            return next;
        }
        public void setNext(Node<E> n) {
            next = n;
        }
    }

    private Node<E> top = null;
    private int size = 0;

    public int size() {
        return size;
    }

    public boolean isEmpty() {
        return size == 0;
    }

    public E top() {
        if (isEmpty()) throw new EmptyStackException();
        return top.getElement();
    }

    public void push(E e) {
        top = new Node<>(e, top);
        size++;
    }

    public E pop() {
        if (isEmpty()) throw new EmptyStackException();
        E answer = top.getElement();
        top = top.getNext();
        size--;
        return answer;
    }

    public void removeBottomHalf() {
        if (isEmpty()) throw new EmptyStackException();
        int halfSize = size / 2;
        Node<E> current = top;
        for (int i = 0; i < halfSize - 1; i++) {
            current = current.getNext();
        }
        current.setNext(null);
        size = halfSize;
    }

    public static void main(String[] args) {
        LinkedStack<Integer> stack = new LinkedStack<>();
        for (int i = 1; i <= 10; i++) {
            stack.push(i);
        }
        System.out.println("Original stack:");
        while (!stack.isEmpty()) {
            System.out.print(stack.pop() + " ");
        }

        for (int i = 1; i <= 10; i++) {
            stack.push(i);
        }
        stack.removeBottomHalf();
        System.out.println("\nStack after removing bottom half:");
        while (!stack.isEmpty()) {
            System.out.print(stack.pop() + " ");
        }
    }
}
```



C:\Windows\System32\cmd.exe

```
Microsoft Windows [Version 10.0.19045.4291]
(c) Microsoft Corporation. All rights reserved.

C:\Users\n1909\OneDrive\Desktop\EduBot\Lab-7\Codes>javac LinkedStack.java

C:\Users\n1909\OneDrive\Desktop\EduBot\Lab-7\Codes>java LinkedStack.java
Original stack:
10 9 8 7 6 5 4 3 2 1
Stack after removing bottom half:
10 9 8 7 6
C:\Users\n1909\OneDrive\Desktop\EduBot\Lab-7\Codes>
```