|  |  |
| --- | --- |
| Student Name | Uzair Hussain |
| Roll Number | 21SW085 |
| Section # | 3rd or III |
| Lab # | 9th – Stack using Array and LinkedList |

**Task#01**

**Code:**

import java.util.NoSuchElementException;

class ArrayStack<T> {

    private T[] stack;

    private int peek;       // peek is the index number

    private int capacity;

    public ArrayStack(int capacity) {

        this.capacity = capacity;

        stack = (T[]) new Object[capacity];     // we can not create an array of generic class so we type casted here

        peek = -1;

    }

    public void push(T data) {

        if (peek == capacity - 1) {

            throw new IllegalStateException("Stack is full");

        }

        stack[++peek] = data;

        peek++;

    }

    public T pop() {

        if (isEmpty()) {

            throw new NoSuchElementException("Stack is empty");

        }

        return stack[peek--];

    }

    public T peek() {

        if (isEmpty()) {

            throw new NoSuchElementException("Stack is empty");

        }

        return stack[peek];

    }

    public boolean isEmpty() {

        return peek == -1;

    }

    public int search(T data) {

        for (int i = peek; i >= 0; i--) {

            if (stack[i].equals(data)) {

                return peek - i + 1;

            }

        }

        return -1;

    }

}

**Output 1:**

**A screenshot of a computer

Description automatically generated with medium confidence**

**Task 2:**

**Code:**

class LinkedStack<T> {

    private Node<T> head;

    private Node pNode;

    private int size;

    private class Node<T> {

        T data;

        Node<T> next;

        public Node(T data) {

            this.data = data;

            this.next = null;

        }

    }

    public LinkedStack() {

        head = null;

    }

    public void push(T data) {

        Node<T> newNode = new Node<>(data);

        newNode.next = head;

        head = newNode;

    }

        public T pop() {

            if (isEmpty()) {

                System.out.println("Stack is empty");

            }

            T data = head.data;

            head = head.next;

            return data;

        }

        public T peek() {

            if (isEmpty()) {

                System.out.println("Stack is empty");

            }

            return head.data;

        }

        public boolean isEmpty() {

            return head == null;

        }

        public int search(T data) {

            int position = 0;

            Node<T> current = head;

            for(current=head;current!=null;current=current.next){

                if(current.data.equals(data)){

                    return position;

                }

                position++;

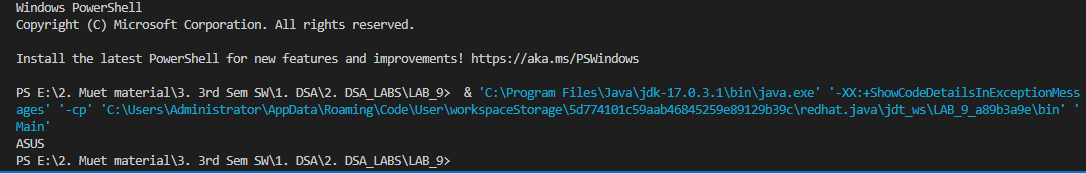
            }

            return -1;

        }

    }

**Output:**

****

**Main Class:**

public class Main{

    public static void main(String[] args) {

        // LinkedStack list=new LinkedStack<>();

        // list.push("HP");

        // list.push("Dell");

        // list.push("ASUS");

        // System.out.println(list.peek());

        ArrayStack a=new ArrayStack<>(3);

        a.push("Apple");

        a.push("Google");

        a.push("Microsoft");

        System.out.println(a.peek());

    }

}

**Github Repository for all Lab Tasks: (from lab 1 to continue)**

[**https://github.com/UzairHussain193/DSA\_LABS\_21SW**](https://github.com/UzairHussain193/DSA_LABS_21SW)

**The End!**