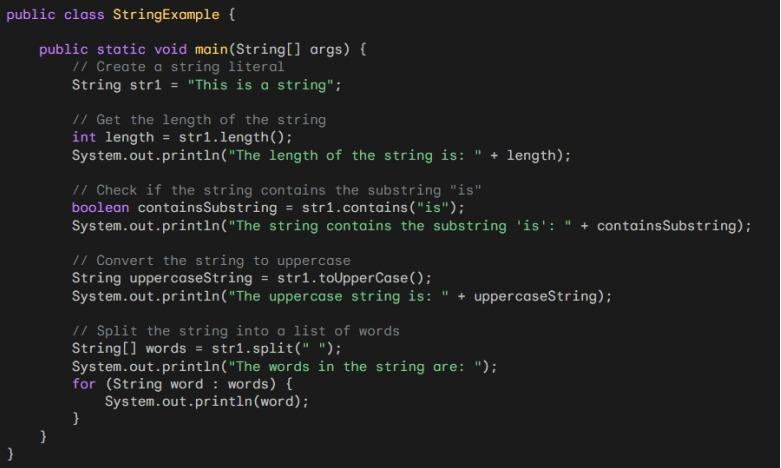
**Java Assignment**

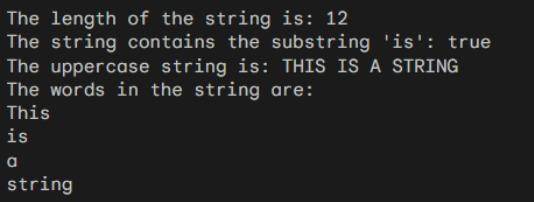
**Name:** Vivek Vardhan

**Registration No:** 21BAI10029

1) String Class In Java:

**Code :  
  
**

**Output:**

  
  
**2) One Dimensional Array In Java  
  
Code:  
  
import java.util.Scanner;**

**public class Example4 {**

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

**try (// creating object of Scanner class**

**Scanner scan = new Scanner(System.in)) {**

**System.out.println("Enter length of Array: ");**

**int arrLength = scan.nextInt();**

**int[] anArray = new int[arrLength];**

**System.out.println("Enter the elements of the Array");**

**for (int i = 0; i < arrLength; i++) {**

**// taking array input**

**anArray[i] = scan.nextInt();**

**}**

**System.out.println("One dimensional array elements are:");**

**for (int i = 0; i < arrLength; i++) {**

**// printing array elements**

**System.out.print(anArray[i] + " ");**

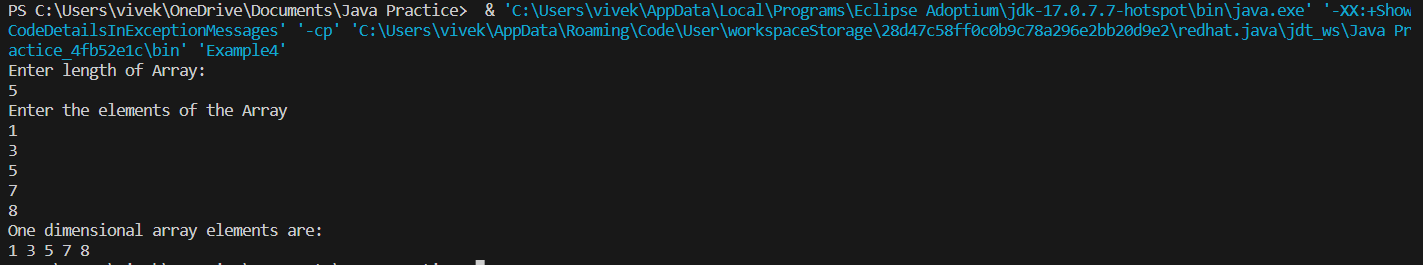
**}**

**}**

**}**

**}**

**Output:**

****

**3) Jagged Array In Java  
  
Code:  
  
import java.util.Scanner;**

**public class Example4 {**

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

**Scanner scan = new Scanner(System.in);**

**System.out.print("Enter the number of sub-arrays: ");**

**int numberOfArrays = scan.nextInt();**

**// Declare the jagged array**

**int[][] jaggedArray = new int[numberOfArrays][];**

**// Allocate memory to each sub-array**

**for (int i = 0; i < numberOfArrays; i++) {**

**System.out.print("Enter the size of sub-array " + (i + 1) + ": ");**

**int sizeOfSubArray = scan.nextInt();**

**jaggedArray[i] = new int[sizeOfSubArray];**

**}**

**// Initialize the elements of each sub-array**

**for (int i = 0; i < numberOfArrays; i++) {**

**System.out.println("Enter the elements of sub-array " + (i + 1) + ":");**

**for (int j = 0; j < jaggedArray[i].length; j++) {**

**jaggedArray[i][j] = scan.nextInt();**

**}**

**}**

**// Print the elements of the jagged array**

**System.out.println("The jagged array is:");**

**for (int i = 0; i < numberOfArrays; i++) {**

**for (int j = 0; j < jaggedArray[i].length; j++) {**

**System.out.print(jaggedArray[i][j] + " ");**

**}**

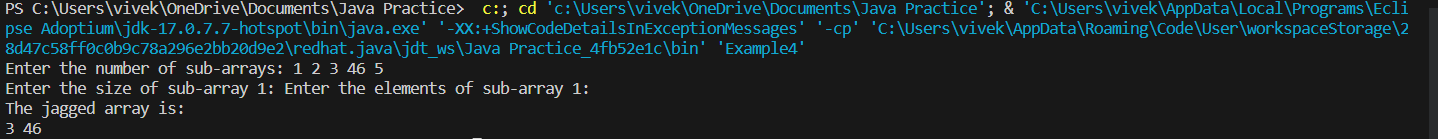
**System.out.println();**

**}**

**scan.close();**

**}**

**}**

**Output:  
  
  
  
4) Collections in Java  
  
Code:  
  
import java.util.\*;**

**class CollectionDemo {**

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

**{**

**// Creating instances of the array,**

**// vector and hashtable**

**int arr[] = new int[] { 1, 2, 3, 4 };**

**Vector<Integer> v = new Vector();**

**Hashtable<Integer, String> h = new Hashtable();**

**// Adding the elements into the**

**// vector**

**v.addElement(1);**

**v.addElement(2);**

**// Adding the element into the**

**// hashtable**

**h.put(1, "geeks");**

**h.put(2, "4geeks");**

**// Array instance creation requires [],**

**// while Vector and hastable require ()**

**// Vector element insertion requires addElement(),**

**// but hashtable element insertion requires put()**

**// Accessing the first element of the**

**// array, vector and hashtable**

**System.out.println(arr[0]);**

**System.out.println(v.elementAt(0));**

**System.out.println(h.get(1));**

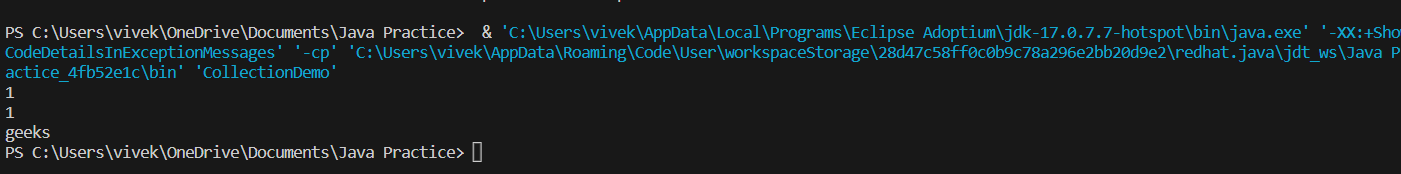
**// Array elements are accessed using [],**

**// vector elements using elementAt()**

**// and hashtable elements using get()**

**}**

**}**

**Output:  
  
5) Byte Stream in Java  
  
Code:  
  
import java.io.\*;**

**public class Example4**

**{**

**public static void main(String[] args) throws IOException**

**{**

**FileInputStream sourceStream = null;**

**FileOutputStream targetStream = null;**

**try**

**{**

**sourceStream = new FileInputStream("source.txt");**

**targetStream = new FileOutputStream ("destination.txt");**

**// Reading source file using read method**

**// and write to file byte by byte using write method**

**int temp;**

**while ((temp = sourceStream.read()) != -1)**

**targetStream.write((byte)temp);**

**}**

**finally**

**{**

**if (sourceStream != null){**

**sourceStream.close();**

**}**

**if (targetStream != null){**

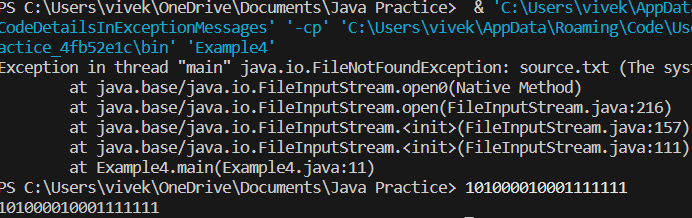
**targetStream.close();**

**}**

**}**

**}**

**}**

**Output:  
  
  
  
6) JDBC Crud Operations**

**Code:  
  
import java.sql.Connection;**

**import java.sql.DriverManager;**

**import java.sql.PreparedStatement;**

**import java.sql.ResultSet;**

**import java.sql.SQLException;**

**/\*\***

**\* Select PreparedStatement JDBC Example**

**\***

**\* @author Ramesh Fadatare**

**\***

**\*/**

**public class Example4 {**

**private static final String QUERY = "select id,name,email,country,password from users where id =?";**

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

**// using try-with-resources to avoid closing resources (boiler plate code)**

**// Step 1: Establishing a Connection**

**try (Connection connection = DriverManager**

**.getConnection("jdbc:mysql://localhost:3306/mysql\_database?useSSL=false", "root", "root");**

**// Step 2:Create a statement using connection object**

**PreparedStatement preparedStatement = connection.prepareStatement(QUERY);) {**

**preparedStatement.setInt(1, 1);**

**System.out.println(preparedStatement);**

**// Step 3: Execute the query or update query**

**ResultSet rs = preparedStatement.executeQuery();**

**// Step 4: Process the ResultSet object.**

**while (rs.next()) {**

**int id = rs.getInt("id");**

**String name = rs.getString("name");**

**String email = rs.getString("email");**

**String country = rs.getString("country");**

**String password = rs.getString("password");**

**System.out.println(id + "," + name + "," + email + "," + country + "," + password);**

**}**

**} catch (SQLException e) {**

**printSQLException(e);**

**}**

**// Step 4: try-with-resource statement will auto close the connection.**

**}**

**public static void printSQLException(SQLException ex) {**

**for (Throwable e: ex) {**

**if (e instanceof SQLException) {**

**e.printStackTrace(System.err);**

**System.err.println("SQLState: " + ((SQLException) e).getSQLState());**

**System.err.println("Error Code: " + ((SQLException) e).getErrorCode());**

**System.err.println("Message: " + e.getMessage());**

**Throwable t = ex.getCause();**

**while (t != null) {**

**System.out.println("Cause: " + t);**

**t = t.getCause();**

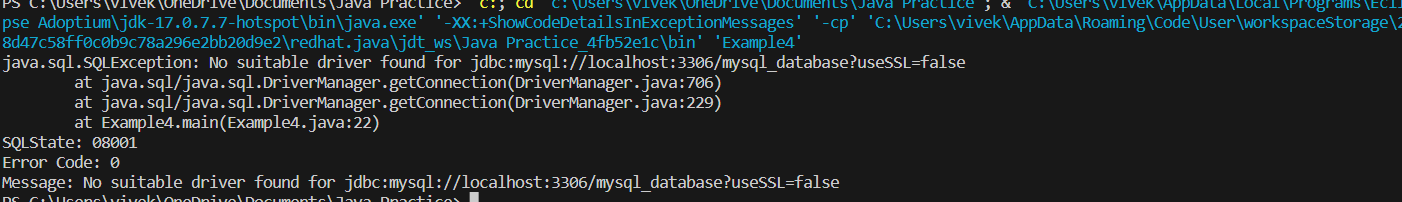
**}**

**}**

**}**

**}**

**}**

**Output:  
  
**

**7) Java Interface  
  
Code:  
  
// Java program to Demonstrate List Interface**

**// Importing all utility classes**

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

**// Main class**

**// ListDemo class**

**class Example4 {**

**// Main driver method**

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

**{**

**// Creating an object of List interface**

**// implemented by the ArrayList class**

**List<Integer> l1 = new ArrayList<Integer>();**

**// Adding elements to object of List interface**

**// Custom inputs**

**l1.add(0, 1);**

**l1.add(1, 2);**

**// Print the elements inside the object**

**System.out.println(l1);**

**// Now creating another object of the List**

**// interface implemented ArrayList class**

**// Declaring object of integer type**

**List<Integer> l2 = new ArrayList<Integer>();**

**// Again adding elements to object of List interface**

**// Custom inputs**

**l2.add(1);**

**l2.add(2);**

**l2.add(3);**

**// Will add list l2 from 1 index**

**l1.addAll(1, l2);**

**System.out.println(l1);**

**// Removes element from index 1**

**l1.remove(1);**

**// Printing the updated List 1**

**System.out.println(l1);**

**// Prints element at index 3 in list 1**

**// using get() method**

**System.out.println(l1.get(3));**

**// Replace 0th element with 5**

**// in List 1**

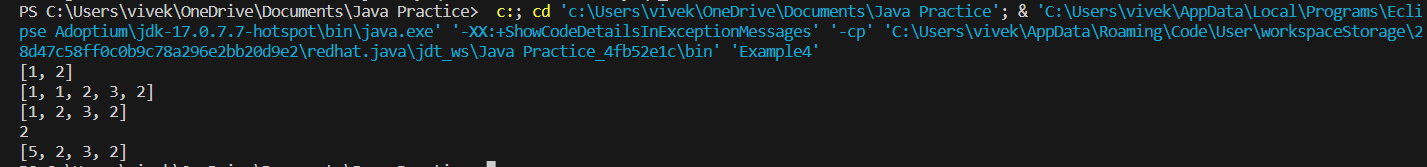
**l1.set(0, 5);**

**// Again printing the updated List 1**

**System.out.println(l1);**

**}**

**}**

**Output:  
  
**

**8) Java Array list**

**Code:**

**// Java program to Demonstrate List Interface**

**// Importing all utility classes**

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

**// Main class**

**// ListDemo class**

**class Example4 {**

**// Main driver method**

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

**{**

**// Creating an object of List interface**

**// implemented by the ArrayList class**

**List<Integer> l1 = new ArrayList<Integer>();**

**// Adding elements to object of List interface**

**// Custom inputs**

**l1.add(0, 1);**

**l1.add(1, 2);**

**// Print the elements inside the object**

**System.out.println(l1);**

**// Now creating another object of the List**

**// interface implemented ArrayList class**

**// Declaring object of integer type**

**List<Integer> l2 = new ArrayList<Integer>();**

**// Again adding elements to object of List interface**

**// Custom inputs**

**l2.add(1);**

**l2.add(2);**

**l2.add(3);**

**// Will add list l2 from 1 index**

**l1.addAll(1, l2);**

**System.out.println(l1);**

**// Removes element from index 1**

**l1.remove(1);**

**// Printing the updated List 1**

**System.out.println(l1);**

**// Prints element at index 3 in list 1**

**// using get() method**

**System.out.println(l1.get(3));**

**// Replace 0th element with 5**

**// in List 1**

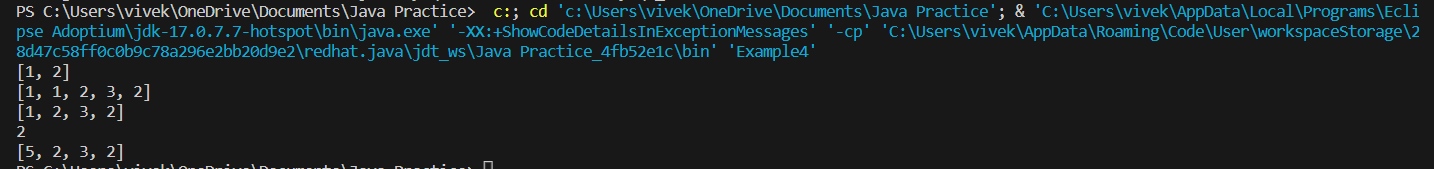
**l1.set(0, 5);**

**// Again printing the updated List 1**

**System.out.println(l1);**

**}**

**}**

**Output:  
  
  
  
9) Java Vector  
  
Code:  
  
import java.util.\*;**

**// Main class**

**class Example4 {**

**// Main driver method**

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

**{**

**// Size of the Vector**

**int n = 5;**

**// Declaring the Vector with**

**// initial size n**

**Vector<Integer> v = new Vector<Integer>(n);**

**// Appending new elements at**

**// the end of the vector**

**for (int i = 1; i <= n; i++)**

**v.add(i);**

**// Printing elements**

**System.out.println(v);**

**// Remove element at index 3**

**v.remove(3);**

**// Displaying the vector**

**// after deletion**

**System.out.println(v);**

**// iterating over vector elements**

**// using for loop**

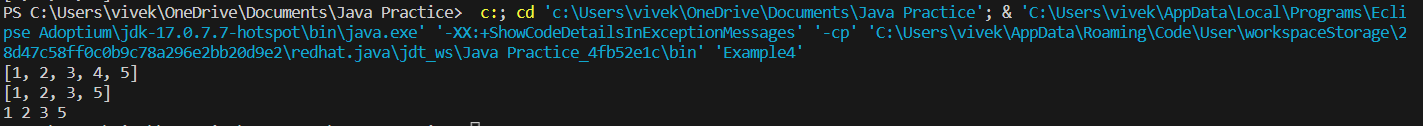
**for (int i = 0; i < v.size(); i++)**

**// Printing elements one by one**

**System.out.print(v.get(i) + " ");**

**}**

**}**

**Output:  
  
  
  
10) Java Stack  
  
Code:  
  
import java.util.\*;**

**class Example4**

**{**

**// Pushing element on the top of the stack**

**static void stack\_push(Stack<Integer> stack)**

**{**

**for(int i = 0; i < 5; i++)**

**{**

**stack.push(i);**

**}**

**}**

**// Popping element from the top of the stack**

**static void stack\_pop(Stack<Integer> stack)**

**{**

**System.out.println("Pop Operation:");**

**for(int i = 0; i < 5; i++)**

**{**

**Integer y = (Integer) stack.pop();**

**System.out.println(y);**

**}**

**}**

**// Displaying element on the top of the stack**

**static void stack\_peek(Stack<Integer> stack)**

**{**

**Integer element = (Integer) stack.peek();**

**System.out.println("Element on stack top: " + element);**

**}**

**// Searching element in the stack**

**static void stack\_search(Stack<Integer> stack, int element)**

**{**

**Integer pos = (Integer) stack.search(element);**

**if(pos == -1)**

**System.out.println("Element not found");**

**else**

**System.out.println("Element is found at position: " + pos);**

**}**

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

**{**

**Stack<Integer> stack = new Stack<Integer>();**

**stack\_push(stack);**

**stack\_pop(stack);**

**stack\_push(stack);**

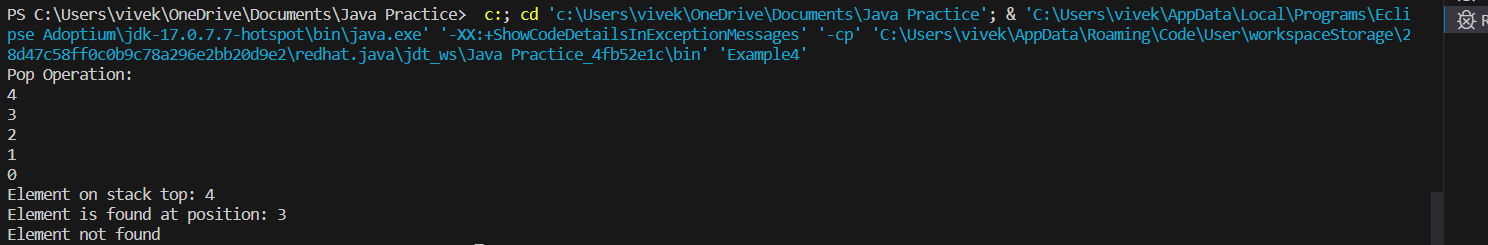
**stack\_peek(stack);**

**stack\_search(stack, 2);**

**stack\_search(stack, 6);**

**}**

**}**

**Output:  
**