**RAMNIRANJAN JHUNJHUNWALA COLLEGE GHATKOPAR (W), MUMBAI - 400 086**

**DEPARTMENT OF INFORMATION TECHNOLOGY**

**2022 - 2023**

**s.Y. B. Sc.( I.T.) SEM iv**

**Paper RJSUITp401 – Advanced Java**

**Name :- Rajbhar Sudesh Dinesh SushilaDevi**

**Roll No. 3067**

**Hindi Vidya Prachar Samiti’s**

**RAMNIRANJAN JHUNJHUNWALA COLLEGE**

**Ghatkopar (W), Mumbai-400 086**

***Certificate***



**This is to certify that Mr./Ms. Rajbhar Sudesh Dinesh SushilaDevi Roll No 3067 of S.Y.B.Sc.(I.T.) class has completed the required number of experiments in the subject of Advanced Java in the Department of Information Technology during the academic year 2022 - 2023 .**

**Professor In-Charge Co-ordinator of IT Department**

**Prof. Bharati Bhole**

**Prof. Archana Bhide**

**College Seal & Date Examiner**

Advanced Java Practicals Journal

# INDEX

Advanced Java Practical Index

| Sr No | Title | Date | Remark |
| --- | --- | --- | --- |
| 1. | **Applet Programming** | 6/1/2023 | DONE |
| 2. | **GUI Programming** | 13/1/2023 | DONE |
| a. | GUI application to display factorial of a number |  |  |
| b. | String Operations - reverse of a string |  |  |
| 3. | **Layouts and Event Handling** |  | DONE |
| a. | AWT applications to demonstrate different layouts | 20/1/2023 |  |
| b. | Write a java program to implement -  MouseEvents  WindowsEvents | 20/1/2023 |  |
| 4. | **Servlet Programming** |  | DONE |
| a. | Create HelloWorld Servlet Application | 21/1/2023 |  |
| b. | Simple Calculator | 21/1/2023 |  |
| c. | Login Page using Servlet | 21/1/2023 |  |
| 5. | **Servlet with Cookies** |  | DONE |
| a. | Servlet with Cookies  Design a servlet with textboxes for accepting Cookie Name and Cookie Value. On click of a submit button, display the values on the same servlet in a table. | 23/2/23 |  |
| 6. | **JSP Programs** |  | DONE |
| a. | Simple JSP Application - Hello World  (Explain the components) | 28/2/23 |  |
| b. | Requestdispatcher JSP -  Design a jsp page to call servlet - welcome servlet and then hello servlet to display data accepted on jsp page. | 28/2/23 |  |
| c. | JSP using intrinsic objects - out, request, response, exception etc.  JSP with validations | 28/2/23 |  |
| c. | JSP to show number of visits  Design jsp application to display number of visits on the website using session | 28/2/23 |  |
| 7. | **Java Database Connectivity and JSP** |  | DONE |
| a. | JSP to demonstrate database connectivity  Fetch student records  Insert records | 16/2/23 |  |
| b. | Servlet to demonstrate login with validations and using databases. | 16/2/23 |  |
| **8.** | **Networking in Java** |  | DONE |
| a. | Demonstrate TCP | 23/2/23 |  |
| b. | Demonstrate UDP | 23/2/23 |  |
| 9. | **Hibernate Application** | 23/2/23 |  |

## Practical 1 - **Applet Programming**

**Applet:** An Applet is a Java program that is embedded in an HTML page that runs on a browser.

#### 1A.Write a Java program to demonstrate an applet.

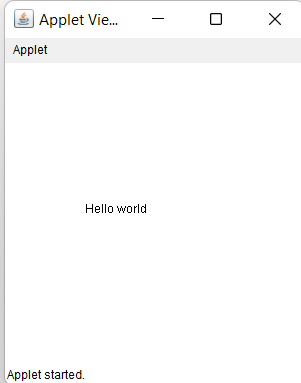
Java file-hello.java

| import java.applet.Applet;  import java.awt.Graphics;  public class hello extends Applet{  public void paint(Graphics g){  g.drawString("Hello world", 80, 150);  }  } |
| --- |

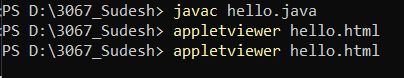
Html file:hello.html

| <!DOCTYPE html>  <html>  <head>  <title>Applet</title>  </head>  <body>  <applet code = "hello.class" width = "300" height = "300"> </applet>  </body>  </html> |
| --- |

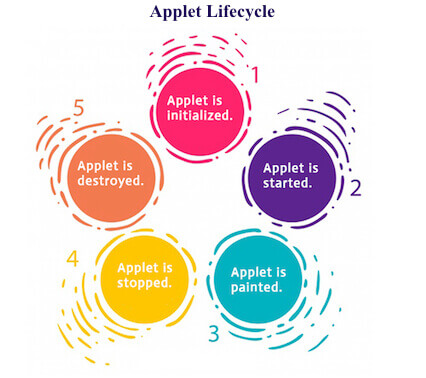
Output:



Commands:



B.Develop an applet to demonstrate the life cycle of java.



1. init() :- It Initialized the Applet. It is called after the param HTML tag (<param> </param>) inside the applet HTML tag has been processed.

Syntax

public void init(){

// rest of the code}

1. start() :- It automatically called after the browser called the init method.

Syntax

public void start(){

// rest of the code}

1. paint() :- It is invoked immediately after the start method and also anytime the applet needs to repaint itself in the browser. It require object of Graphics Class not mandatory all the time.

Syntax

public void paint(Graphics g){

// rest of the code}

.

1. stop() :- It automatically calls when the user moves off the page or window on which the applet sits.

Syntax

public void stop(){

// rest of the code}

1. destroy() :- It is called when the browser shuts down normally.

Syntax

public void destroy(){

// rest of the code

}

* Using paint and init to change the background color.

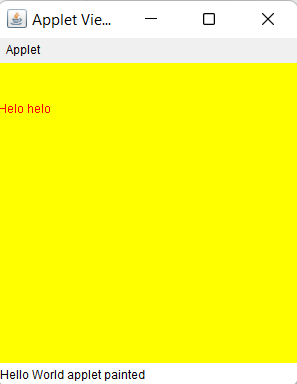
Java File- hello1.java

| import java.applet.Applet;  import java.awt.\*;  public class hello1 extends Applet{  public void init() {  setBackground(Color.yellow);  }  public void paint(Graphics g) {  g.setColor(Color.red);  g.drawString("Helo helo", 0, 50);  showStatus("Hello World applet painted");  }  } |
| --- |

Htmlfile-hello1.html

| <!DOCTYPE html>  <html>  <head>  <title>Applet</title>  </head>  <body>  <applet code = "hello1.class" width = "300" height = "300"> </applet>  </body>  </html> |
| --- |

Output:



Command:



## 

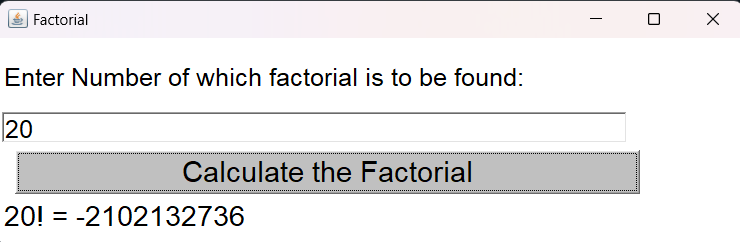
## **Practical 2 - GUI Programming**

#### 2A.Design a AWT program to print the factorial for an input value.

FileName:awtfact.java

| import java.awt.\*;  import java.awt.event.\*;  public class awtfact implements ActionListener{  Frame frame;  Label lb1, lb2;  TextField tf1;  Button b1;  public awtfact() {  frame = new Frame("Factorial");  lb1 = new Label("Enter Number of which factorial is to be found:");  tf1 = new TextField();  b1 = new Button("Calculate the Factorial");  lb2 = new Label();  lb1.setBounds(10, 50, 500, 25);  tf1.setBounds(10, 90, 500, 25);  b1.setBounds(20, 120, 500, 35);  lb2.setBounds(10, 160,500, 25);  /\* lb1.setBounds(10, 50, 140, 25);  tf1.setBounds(150, 50, 120, 25);  b1.setBounds(20, 90, 250, 35);  lb2.setBounds(00, 140, 300, 25);  \*/    lb1.setFont(new Font("Calibiri", 0, 20));  tf1.setFont(new Font("TimesRoman", 0, 20));  b1.setFont(new Font("TimesRoman", 0, 23));  lb2.setFont(new Font("TimesRoman", 0, 22));    lb1.setAlignment(Label.LEFT);  lb2.setAlignment(Label.LEFT);    b1.setBackground(Color.LIGHT\_GRAY);  frame.add(b1);  frame.add(lb1);  frame.add(tf1);  frame.add(lb2);  frame.setSize(500, 300);  frame.setResizable(true);  frame.setLayout(null);  frame.setVisible(true);  b1.addActionListener(this);  frame.addWindowListener(new WindowAdapter() {  public void windowClosing(WindowEvent e) {  frame.dispose();  }  });  }  public void actionPerformed(ActionEvent e) {  try{  frame.setSize(300, 180);  if (e.getSource() == b1) {  int num = Integer.parseInt(tf1.getText());  lb2.setText(num + "! = " + calculateFactorial(num));  }  }  catch(Exception ex){  lb2.setText("Please give Valid input.");  }  }  public int calculateFactorial(int n) {  int fact = 1, i;  for (i = n; i > 0; i--) {  fact = fact \* i;  }  return fact;  }  public static void main(String[] args) {  awtfact awt = new awtfact();  }  } |
| --- |

Output:

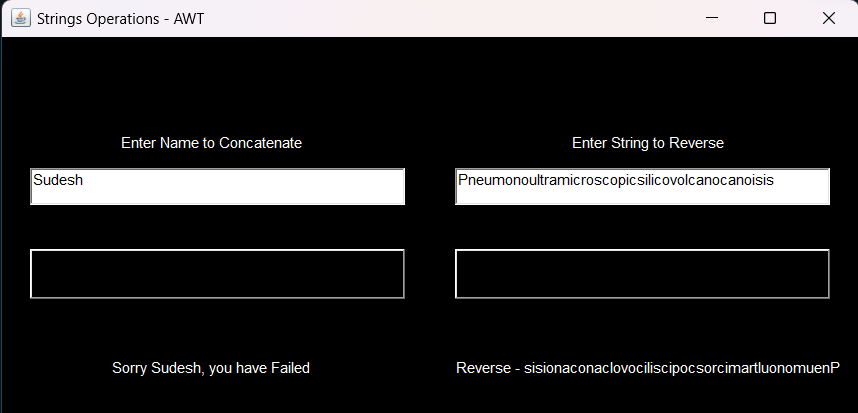


#### 

#### 2B.Design an AWT program to perform various string operations like reverse string,string concatenation etc.

| import java.awt.\*;  import java.awt.event.\*;  public class StringAWT implements ActionListener {  Frame frame;  Panel p1, p2;  Button p1Btn, p2Btn;  Label p1Lb2, p1Lb3, p2Lb2, p2Lb3;  TextField p1Tf, p2Tf;  public StringAWT() {  frame = new Frame("Strings Operations - AWT");  concatPanel();  reversePanel();  frame.setSize(700, 300);  frame.setLayout(null);  frame.setVisible(true);  p1Btn.addActionListener(this);  p2Btn.addActionListener(this);  frame.addWindowListener(new WindowAdapter() {  public void windowClosing(WindowEvent e) {  frame.dispose();  }  });  }  public void concatPanel() {  p1 = new Panel();  p1Lb2 = new Label("Enter Name to Concatenate");  p1Tf = new TextField();  p1Btn = new Button("Concatenate");  p1Lb3 = new Label();  p1.setBounds(0, 0, 350, 350);  p1Lb2.setForeground(Color.WHITE);  p1Lb3.setForeground(Color.WHITE);  p1.setBackground(Color.BLACK);  p1Lb2.setBackground(Color.BLACK);  p1Btn.setBackground(Color.BLACK);  p1Lb3.setBackground(Color.BLACK);  p1Lb2.setBounds(00, 100, 350, 30);  p1Tf.setBounds(30, 135, 300, 30);  p1Btn.setBounds(30, 200, 300, 40);  p1Lb3.setBounds(00, 280, 350, 30);        p1Lb2.setAlignment(Label.CENTER);  p1Lb3.setAlignment(Label.CENTER);  p1.add(p1Lb2);  p1.add(p1Tf);  p1.add(p1Btn);  p1.add(p1Lb3);  frame.add(p1Lb2);  frame.add(p1Tf);  frame.add(p1Btn);  frame.add(p1Lb3);  frame.add(p1);  }  public void reversePanel() {  p2 = new Panel();  p2Lb2 = new Label("Enter String to Reverse");  p2Tf = new TextField();  p2Btn = new Button("Reverse It");  p2Lb3 = new Label();  p2.setBounds(350, 0, 350, 350);  p2Lb2.setForeground(Color.white);  p2Lb2.setBounds(350, 100, 350, 30);  p2Tf.setBounds(370, 135, 300, 30);  p2Btn.setBounds(370, 200, 300, 40);  p2Lb3.setBounds(350, 280, 350, 30);  p2Lb2.setAlignment(Label.CENTER);  p2Lb3.setAlignment(Label.CENTER);    p2.setBackground(Color.BLACK);  p2Lb2.setBackground(Color.BLACK);  p2Btn.setBackground(Color.BLACK);  p2Lb3.setBackground(Color.BLACK);  p2Lb3.setForeground(Color.WHITE);  p2.add(p2Lb2);  p2.add(p2Tf);  p2.add(p2Btn);  p2.add(p2Lb3);  frame.add(p2Lb2);  frame.add(p2Tf);  frame.add(p2Btn);  frame.add(p2Lb3);  frame.add(p2);  }  public void actionPerformed(ActionEvent e) {  frame.setSize(700, 350);  if (e.getSource() == p1Btn) {  p1Lb3.setText("Sorry " + p1Tf.getText() + ", you have Failed");  }  if (e.getSource() == p2Btn) {  java.lang.String str = p2Tf.getText();  char ch[] = str.toCharArray();  java.lang.String reverse = "";  for (int i = ch.length - 1; i >= 0; i--) {  reverse += ch[i];  }  p2Lb3.setText("Reverse - " + reverse);  }  }  public static void main(String[] args) {  StringAWT awtString = new StringAWT();  }  } |
| --- |

Output**:**

****

## Practical 3 - Layouts and Event Handling

#### 3A.Design an AWT application to demonstrate different layouts

##### **1.Flow Layout:**

* The Java FlowLayout class is used to arrange the components in a line, one after another (in a flow).
* Flow layout puts components (such as text fields, buttons, labels etc) in a row, if horizontal space is not enough to hold all components then Flow layout adds them in a next row and so on.
* All rows in Flow layout are center aligned by default.
* The default horizontal and vertical gap between components is 5 pixels.
* FlowLayout class is present in java.awt package.

**Classes Used :-**

1. Flowlayout Class (java.awt.FlowLayout)

**Methods Used :-**

1. FlowLayout() → creates a flow layout with centered alignment and a default 5 unit horizontal and vertical gap.

**Syntax :-**

Flowlayout fl = new Flowlayout();

1. FlowLayout(int align) → creates a flow layout with the given alignment and a default 5 unit horizontal and vertical gap.

**Syntax :-**

Flowlayout fl = new Flowlayout(int align);

**Example :-**

Flowlayout fl1 = new FlowLayout(FlowLayout.LEFT); // FlowLayout.LEFT OR -1

Flowlayout fl2 = new FlowLayout(FlowLayout.CENTER); // FlowLayout.CENTER OR 1

Flowlayout fl3 = new FlowLayout(FlowLayout.RIGHT); // FlowLayout.RIGHT OR 2

1. FlowLayout(int align, int hgap, int vgap) → creates a flow layout with the given alignment and the given horizontal and vertical gap.

**Syntax :-**

Flowlayout fl = new Flowlayout(int align, int hgap, int vgap);

**Example :-**

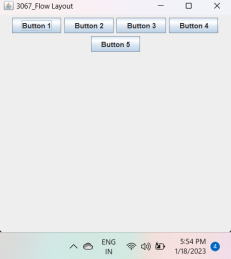
Flowlayout fl1 = new FlowLayout(FlowLayout.LEFT, 40, 80); // FlowLayout.LEFT OR -1

Flowlayout

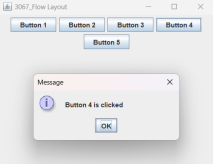
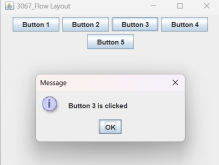
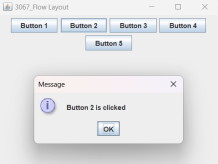
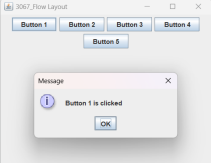
| import java.awt.\*;  import java.awt.event.\*;  import javax.swing.\*;  public class QflowLayout implements ActionListener {  JFrame frame = new JFrame("3067\_Flow Layout");  JButton b1, b2, b3, b4, b5;  FlowLayout flowLay = new FlowLayout();  public QflowLayout() {  b1 = new JButton("Button 1");  b2 = new JButton("Button 2");  b3 = new JButton("Button 3");  b4 = new JButton("Button 4");  b5 = new JButton("Button 5");  b1.addActionListener(this);  b2.addActionListener(this);  b3.addActionListener(this);  b4.addActionListener(this);  b5.addActionListener(this);  frame.add(b1);  frame.add(b2);  frame.add(b3);  frame.add(b4);  frame.add(b5);  frame.setLayout(flowLay);  frame.setSize(400, 400);  frame.setVisible(true);  frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);  }  public void actionPerformed(ActionEvent e) {  if (e.getSource() == b1) {  JOptionPane.showMessageDialog(frame, b1.getText() + " is clicked");  } else if (e.getSource() == b2) {  JOptionPane.showMessageDialog(frame, b2.getText() + " is clicked");  } else if (e.getSource() == b3) {  JOptionPane.showMessageDialog(frame, b3.getText() + " is clicked");  } else if (e.getSource() == b4) {  JOptionPane.showMessageDialog(frame, b4.getText() + " is clicked");  } else if (e.getSource() == b5) {  JOptionPane.showMessageDialog(frame, b5.getText() + " is clicked");  }  }  public static void main(String[] args) {  new QflowLayout();  }  } |
| --- |

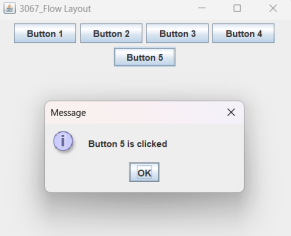
:

oUTPUT:

****

**All Buttons Clicked:**

****

****

##### **2.Border layout:**

* The BorderLayout is used to arrange the components in five regions: north, south, east, west, and center.
* Each region (area) may contain one component only.
* It is the default layout of a frame or window.
* It is present in the java.awt package.

**Classes Used :-**

1. Border Layout Class (java.awt.BorderLayout)

**Methods Used :-**

1. BorderLayout() → creates a border layout but with no gaps between the components.

**Syntax :-**

BorderLayout bl = new BorderLayout();

1. BorderLayout(int hgap, int vgap) → creates a border layout with the given horizontal and vertical gaps between the components.

**Syntax :-**

BorderLayout bl = new BorderLayout(50, 50);

BorderLayout

| import java.awt.\*;  import java.awt.event.\*;  import javax.swing.\*;  public class assignment implements ActionListener {  JFrame frame = new JFrame("Layout");  JButton b1, b2, b3, b4, b5;  BorderLayout borderLay = new BorderLayout();  public assignment() {  b1 = new JButton("Button 1");  b2 = new JButton("Button 2");  b3 = new JButton("Button 3");  b4 = new JButton("Button 4");  b5 = new JButton("Button 5");  b1.addActionListener(this);  b2.addActionListener(this);  b3.addActionListener(this);  b4.addActionListener(this);  b5.addActionListener(this);  frame.add(b1, borderLay.EAST);  frame.add(b2, borderLay.NORTH);  frame.add(b3, borderLay.WEST);  frame.add(b4, borderLay.SOUTH);  frame.add(b5, borderLay.CENTER);  frame.setSize(350, 350);  frame.setVisible(true);  frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);  }  public void actionPerformed(ActionEvent e) {  if (e.getSource() == b1) {  JOptionPane.showMessageDialog(frame, b1.getText() + " is clicked"); } else if (e.getSource() == b2) {  JOptionPane.showMessageDialog(frame, b2.getText() + " is clicked"); } else if (e.getSource() == b3) {  JOptionPane.showMessageDialog(frame, b3.getText() + " is clicked"); } else if (e.getSource() == b4) {  JOptionPane.showMessageDialog(frame, b4.getText() + " is clicked"); } else if (e.getSource() == b5) {  JOptionPane.showMessageDialog(frame, b5.getText() + " is clicked"); |
| --- |

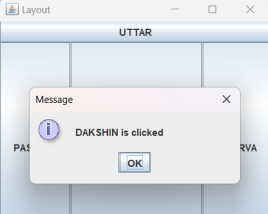
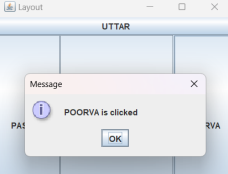
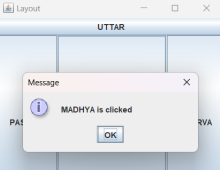
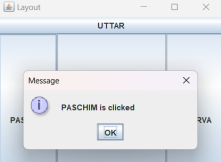
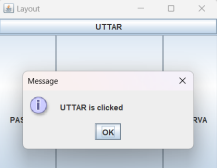
**Advanced Java 3067\_Sudesh Rajbhar**

| }  }  public static void main(String[] args) {  new assignment();  }  } |
| --- |

**Output -**

**Basic:**

****

****

##### 

##### 

##### **3.Grid layout:**

* The Java GridLayout class is used to arrange the components in a rectangular grid.
* One component is displayed in each rectangle.
* GridLayout class is present in java.awt package.

**Classes Used :-**

1. GridLayout Class (java.awt.GridLayout)

**Methods Used :-**

1. GridLayout() → creates a grid layout with one column per component in a row.

**Syntax :-**

GridLayout gl = new GridLayout();

1. GridLayout(int rows, int columns) → creates a grid layout with the given rows and columns but no gaps between the components.

**Syntax :-**

GridLayout gl = new GridLayout(3, 5);

1. GridLayout(int rows, int columns, int hgap, int vgap) → creates a grid layout with the given rows and columns along with given horizontal and vertical gaps.

**Syntax :-**

GridLayout gl = new GridLayout(3, 5, 40, 40);

Grid layout

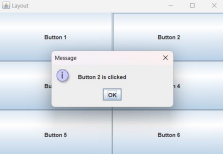
| import java.awt.\*;  import java.awt.event.\*;  import javax.swing.\*;  public class QgridLayout implements ActionListener {  JFrame frame = new JFrame("Layout");  JButton b1, b2, b3, b4, b5 , b6;  GridLayout gridLay = new GridLayout(3,2);  public QgridLayout() {  b1 = new JButton("Button 1");  b2 = new JButton("Button 2");  b3 = new JButton("Button 3");  b4 = new JButton("Button 4");  b5 = new JButton("Button 5");  b6 = new JButton("Button 6");  b1.addActionListener(this);  b2.addActionListener(this);  b3.addActionListener(this);  b4.addActionListener(this);  b5.addActionListener(this);  b6.addActionListener(this);  frame.add(b1);  frame.add(b2);  frame.add(b3);  frame.add(b4);  frame.add(b5);  frame.add(b6);  frame.setLayout(gridLay );  frame.setSize(500, 350);  frame.setVisible(true);  frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);  }  public void actionPerformed(ActionEvent e) {  if (e.getSource() == b1) {  JOptionPane.showMessageDialog(frame, b1.getText() + " is clicked"); } else if (e.getSource() == b2) {  JOptionPane.showMessageDialog(frame, b2.getText() + " is clicked"); } else if (e.getSource() == b3) {  JOptionPane.showMessageDialog(frame, b3.getText() + " is clicked"); |
| --- |

**Advanced Java 3067\_Sudesh Rajbhar**

| } else if (e.getSource() == b4) {  JOptionPane.showMessageDialog(frame, b4.getText() + " is clicked"); } else if (e.getSource() == b5) {  JOptionPane.showMessageDialog(frame, b5.getText() + " is clicked"); }  }  public static void main(String[] args) {  new QgridLayout();  }  } |
| --- |

**Output -**

**Basic**:

****

##### **4.Card Layout:**

* The CardLayout class manages the components in such a way that only one component is visible at a time.
* It treats each component as a card in the container.
* Only one card is visible at a time,and the container acts as a stack of cards
* The first component added to a CardLayout object is the visible component, when the container is first displayed.
* CardLayout class is found in java.awt package.

**Classes Used :-**

1. CardLayout Class (java.awt.CardLayout)

**Methods Used :-**

1. CardLayout() → creates a card layout with zero horizontal and vertical gap.

**Syntax :-**

CardLayout cl = new CardLayout();

1. CardLayout(int hgap, int vgap) → creates a card layout with the given horizontal and vertical gap.

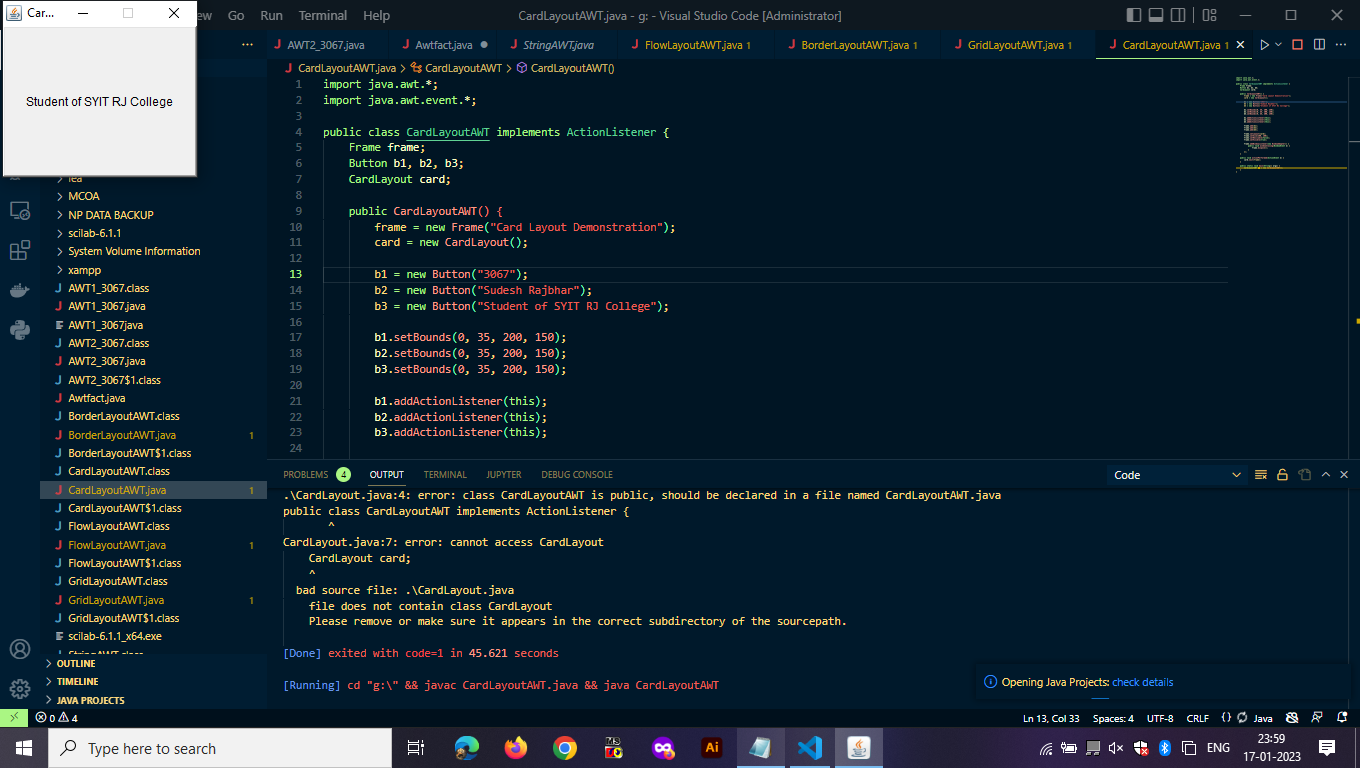
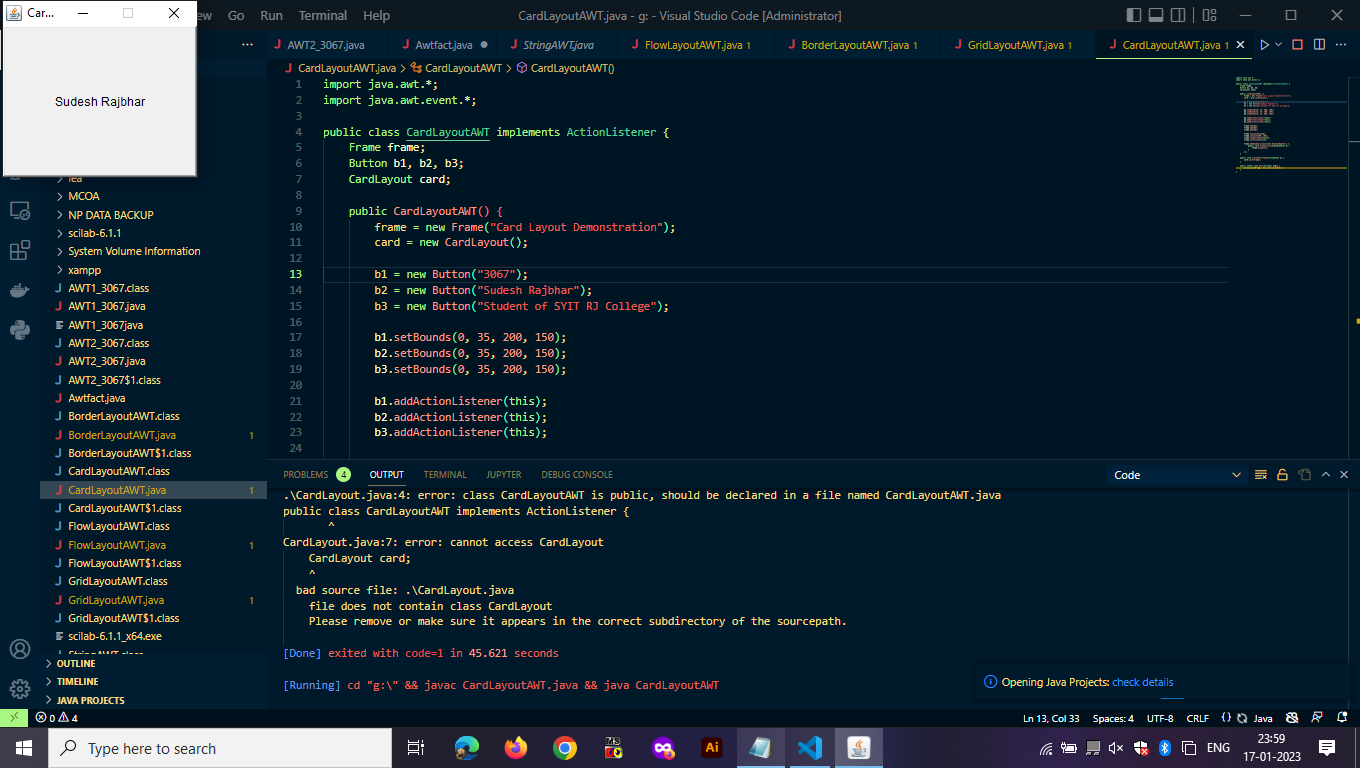
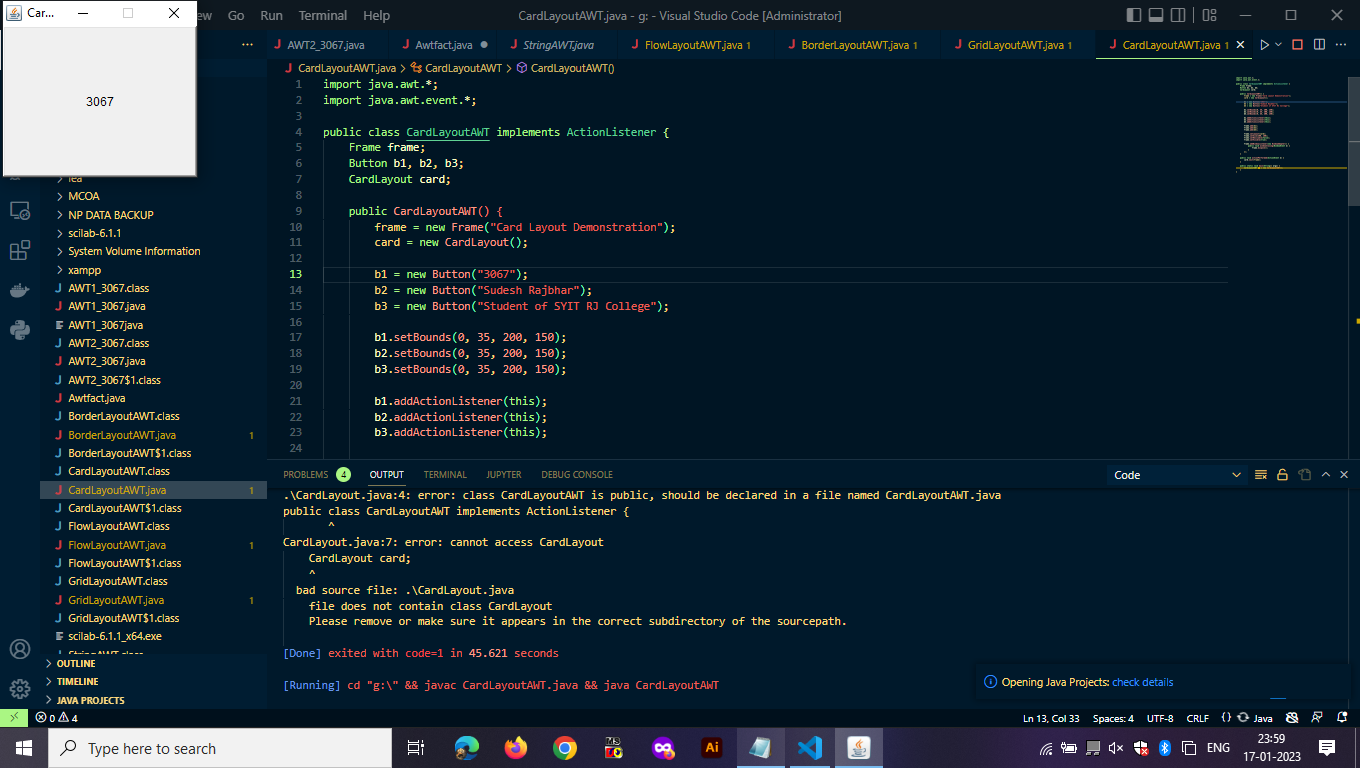
**Syntax :-**

CardLayout cl = new CardLayout(40, 30);

**File name (CardLayoutAWT.java)**

| import java.awt.\*;  import java.awt.event.\*;  public class CardLayoutAWT implements ActionListener {  Frame frame;  Button b1, b2, b3;  CardLayout card;  public CardLayoutAWT() {  frame = new Frame("Card Layout Demonstration");  card = new CardLayout();  b1 = new Button("3067");  b2 = new Button("Sudesh Rajbhar");  b3 = new Button("Student of SYIT RJ College");  b1.setBounds(0, 35, 200, 150);  b2.setBounds(0, 35, 200, 150);  b3.setBounds(0, 35, 200, 150);  b1.addActionListener(this);  b2.addActionListener(this);  b3.addActionListener(this);  frame.add(b1);  frame.add(b2);  frame.add(b3);  frame.setLayout(card);  frame.setSize(200, 180);  frame.setResizable(false);  frame.setVisible(true);  frame.addWindowListener(new WindowAdapter() {  public void windowClosing(WindowEvent e) {  frame.dispose();  }  });  }  public void actionPerformed(ActionEvent e) {  card.next(frame);  }  public static void main(String[] args) {  CardLayoutAWT fl = new CardLayoutAWT();  }  } |
| --- |

**Output:**

****

### 3B)Write a java program to implement –

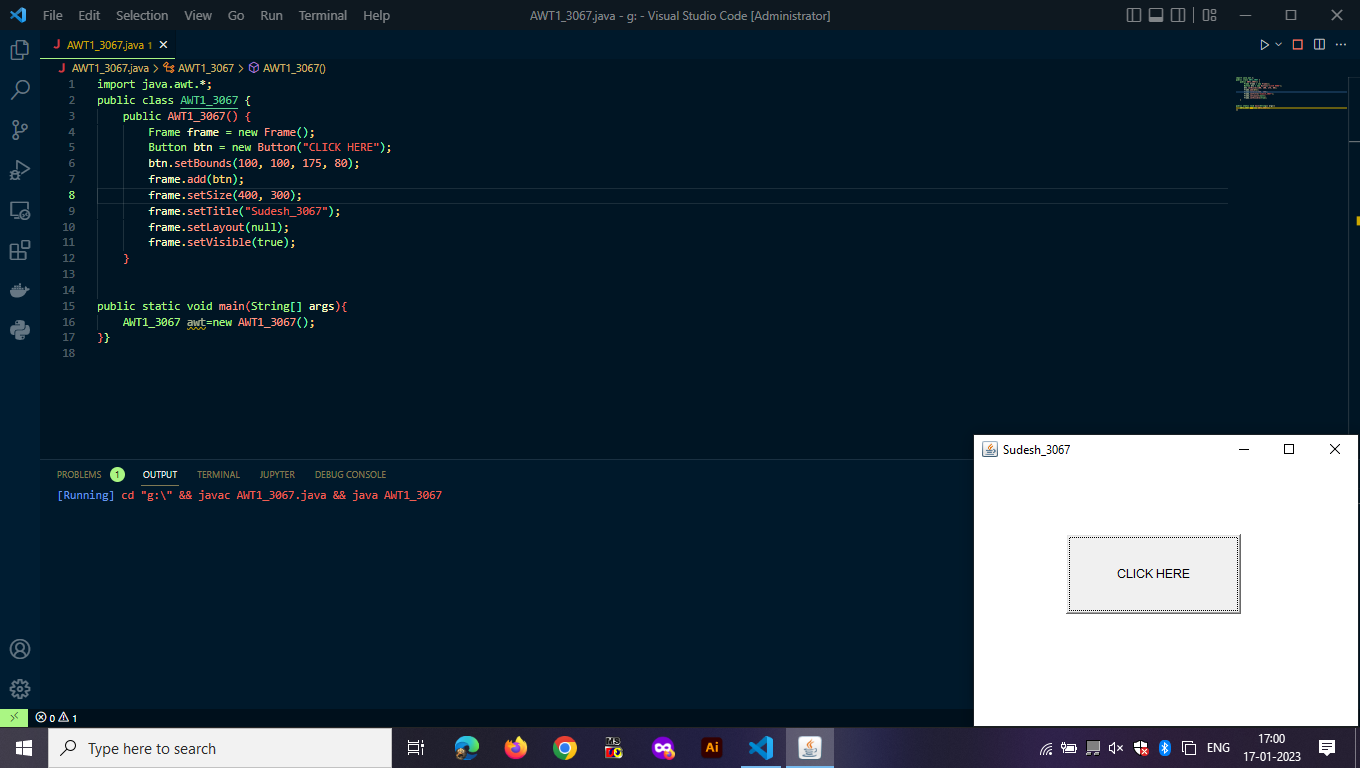
#### Text Events

Filename: AWT1.java

| import java.awt.\*;  public class AWT1\_3067 {  public AWT1\_3067() {  Frame frame = new Frame();  Button btn = new Button("CLICK HERE");  btn.setBounds(100, 100, 175, 80);  frame.add(btn);  frame.setSize(400, 300);  frame.setTitle("Sudesh\_3067");  frame.setLayout(null);  frame.setVisible(true);  }  public static void main(String[] args){  AWT1\_3067 awt=new AWT1\_3067();  }} |
| --- |

Output:





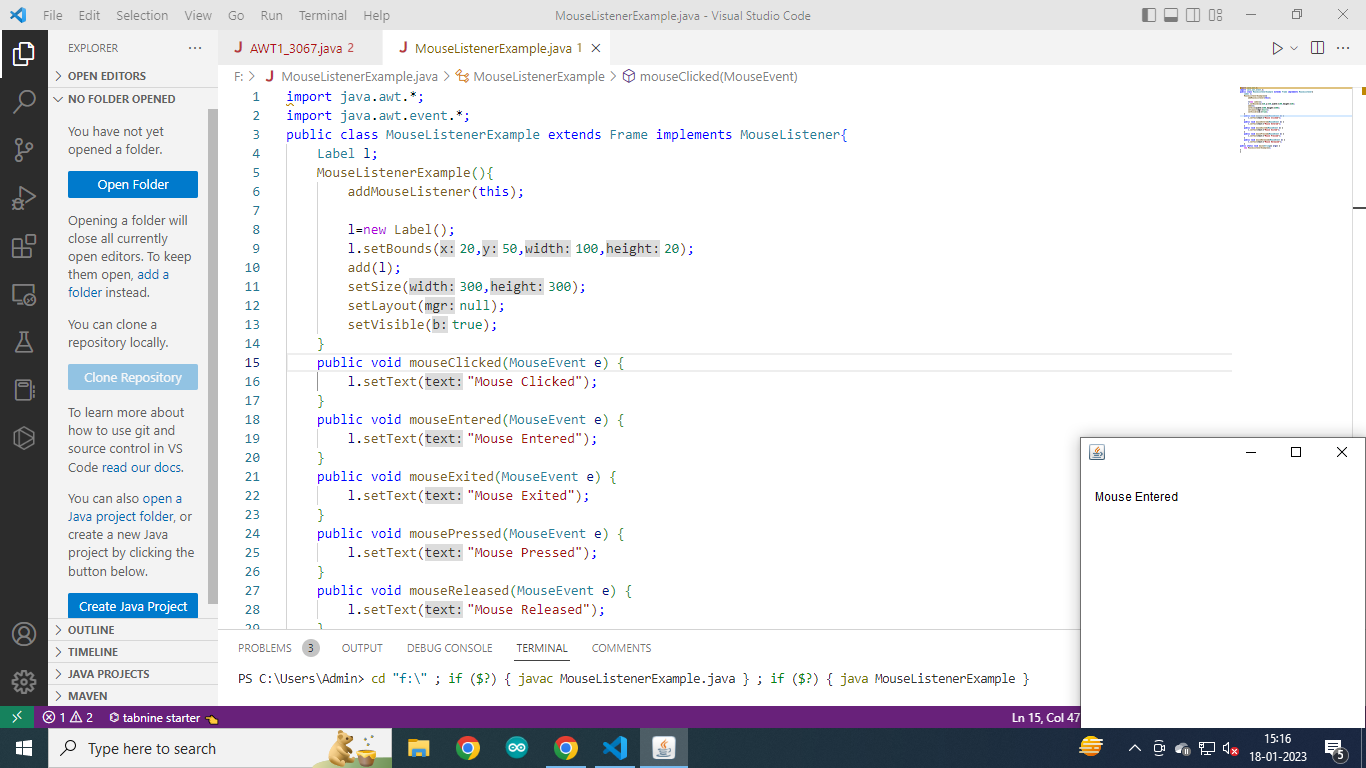
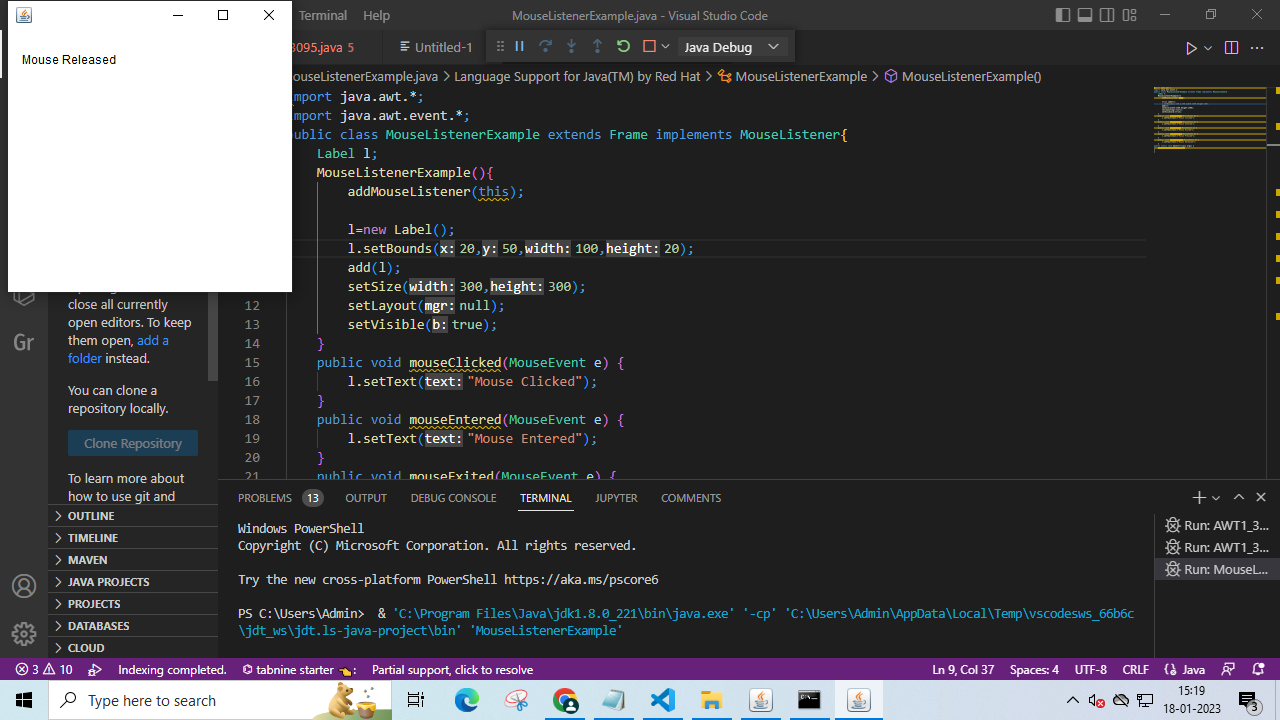
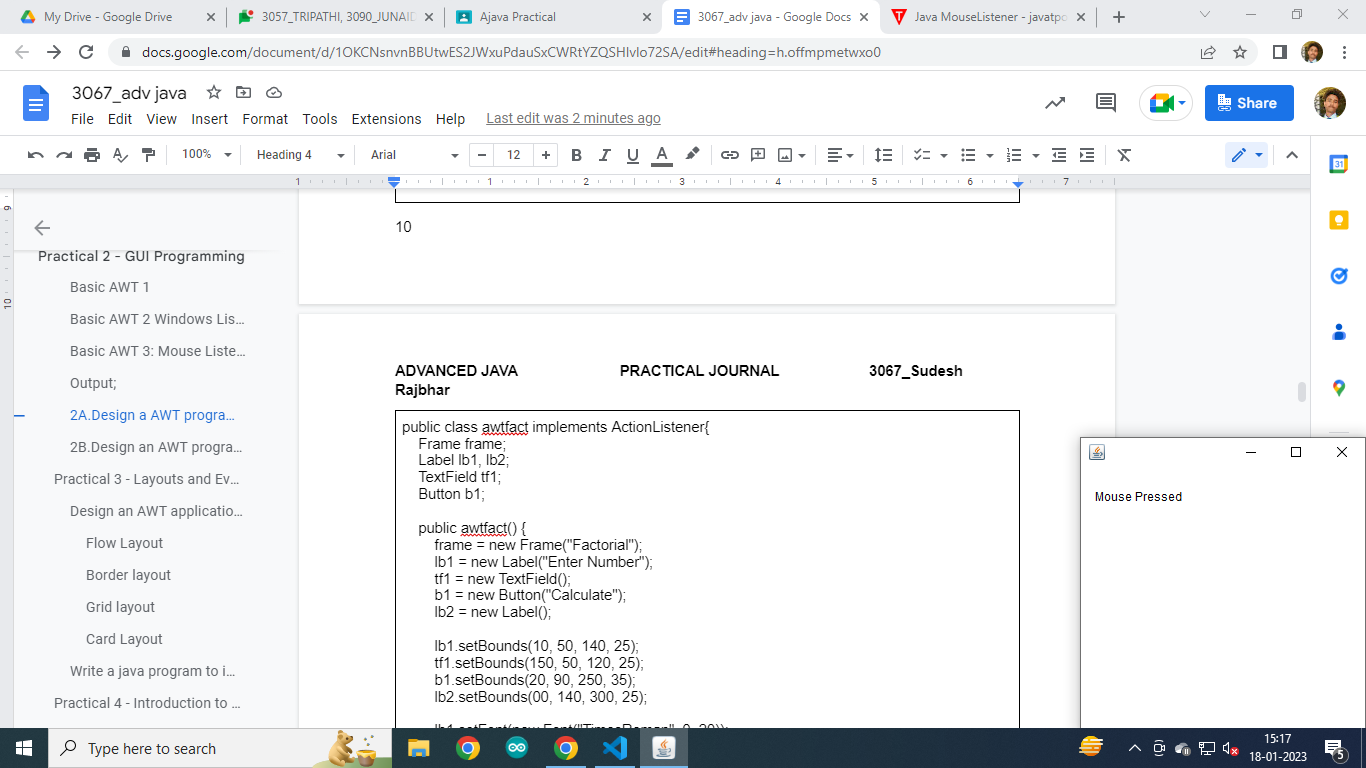
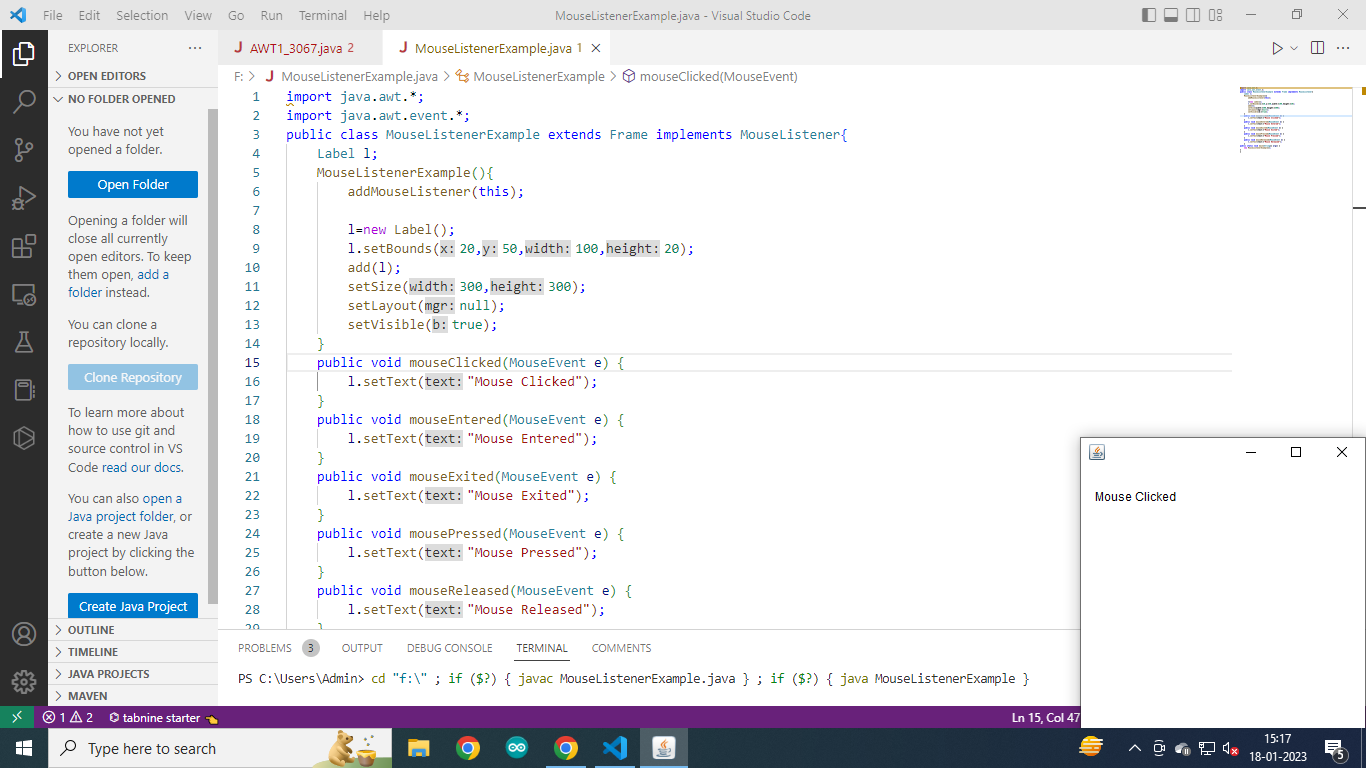
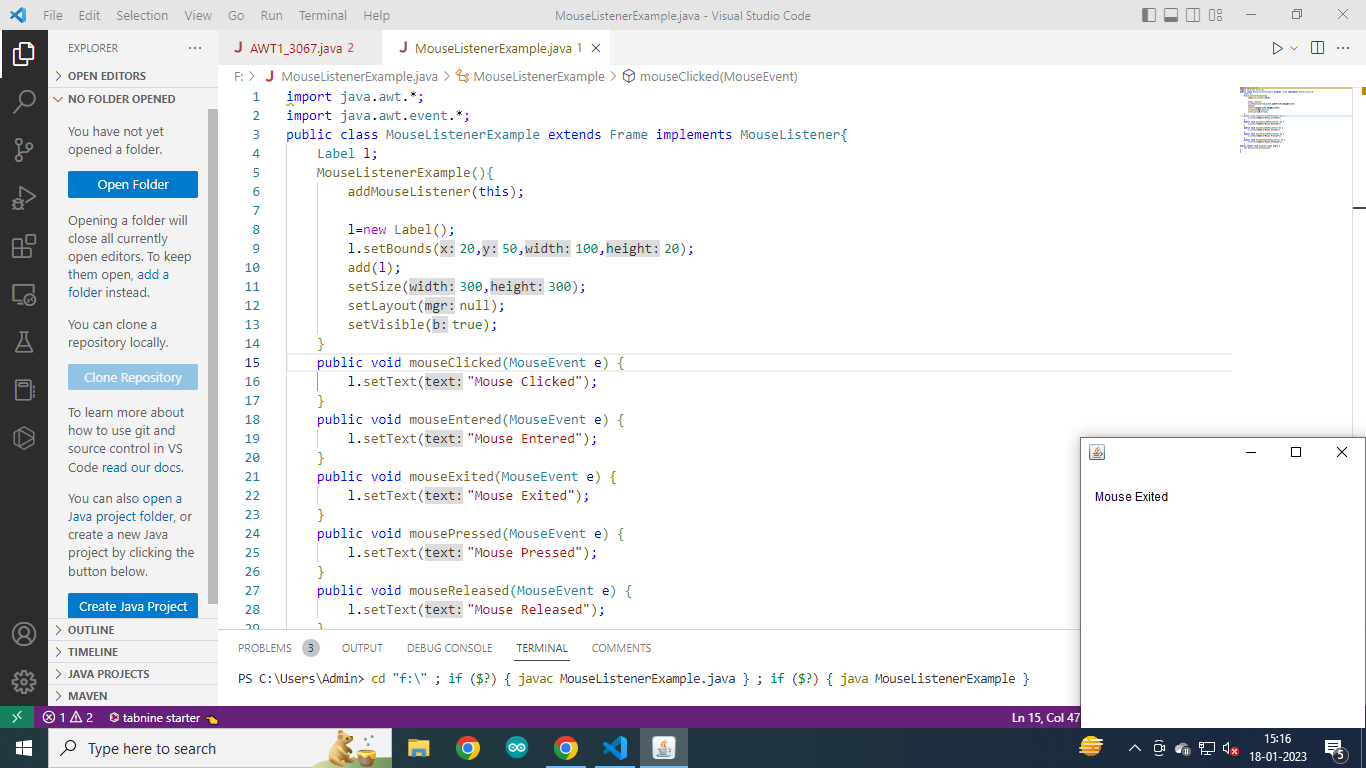
#### Mouse Events

Basic AWT 3: Mouse Listener.

MouseListenerExample.java

| import java.awt.\*;  import java.awt.event.\*;  public class MouseListenerExample extends Frame implements MouseListener{  Label l;  MouseListenerExample(){  addMouseListener(this);    l=new Label();  l.setBounds(20,50,100,20);  add(l);  setSize(300,300);  setLayout(null);  setVisible(true);  }  public void mouseClicked(MouseEvent e) {  l.setText("Mouse Clicked");  }  public void mouseEntered(MouseEvent e) {  l.setText("Mouse Entered");  }  public void mouseExited(MouseEvent e) {  l.setText("Mouse Exited");  }  public void mousePressed(MouseEvent e) {  l.setText("Mouse Pressed");  }  public void mouseReleased(MouseEvent e) {  l.setText("Mouse Released");  }  public static void main(String[] args) {  new MouseListenerExample();  }  } |
| --- |

Output;



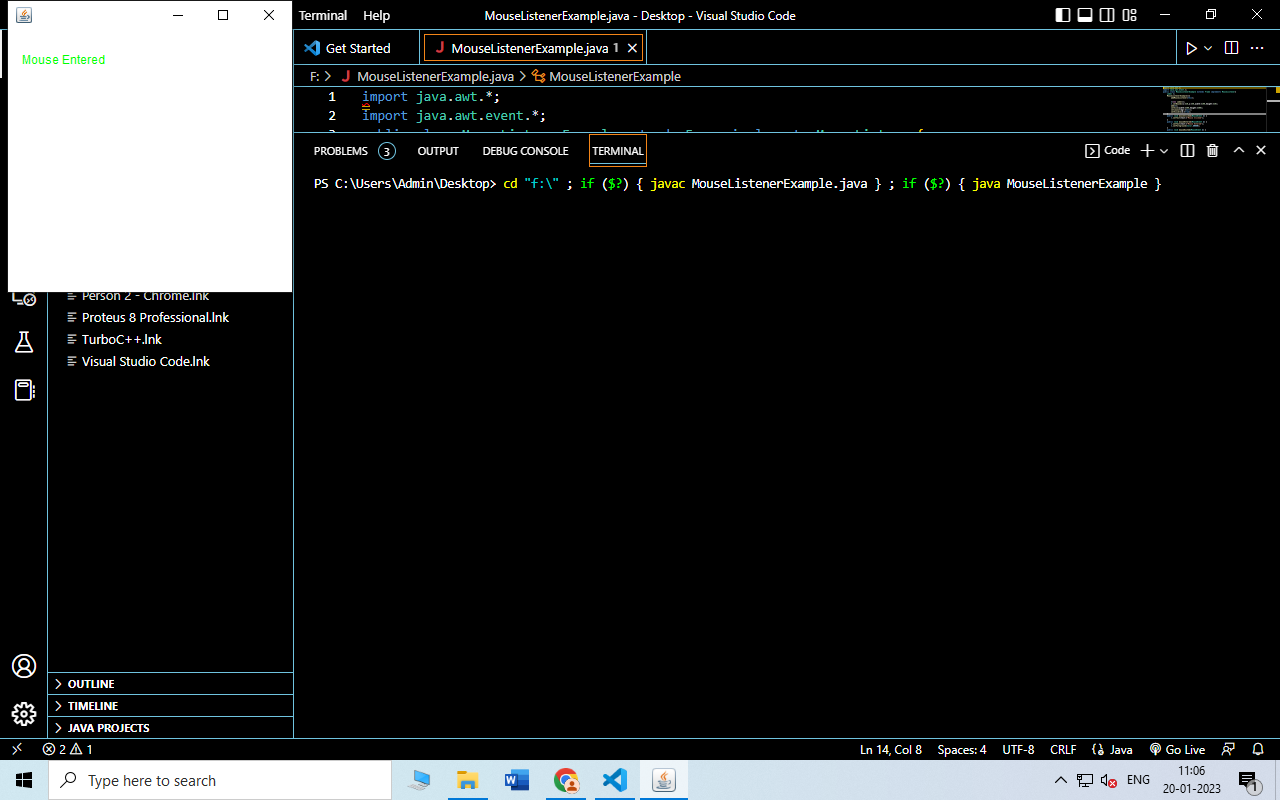
Mouse Listener:

Code:

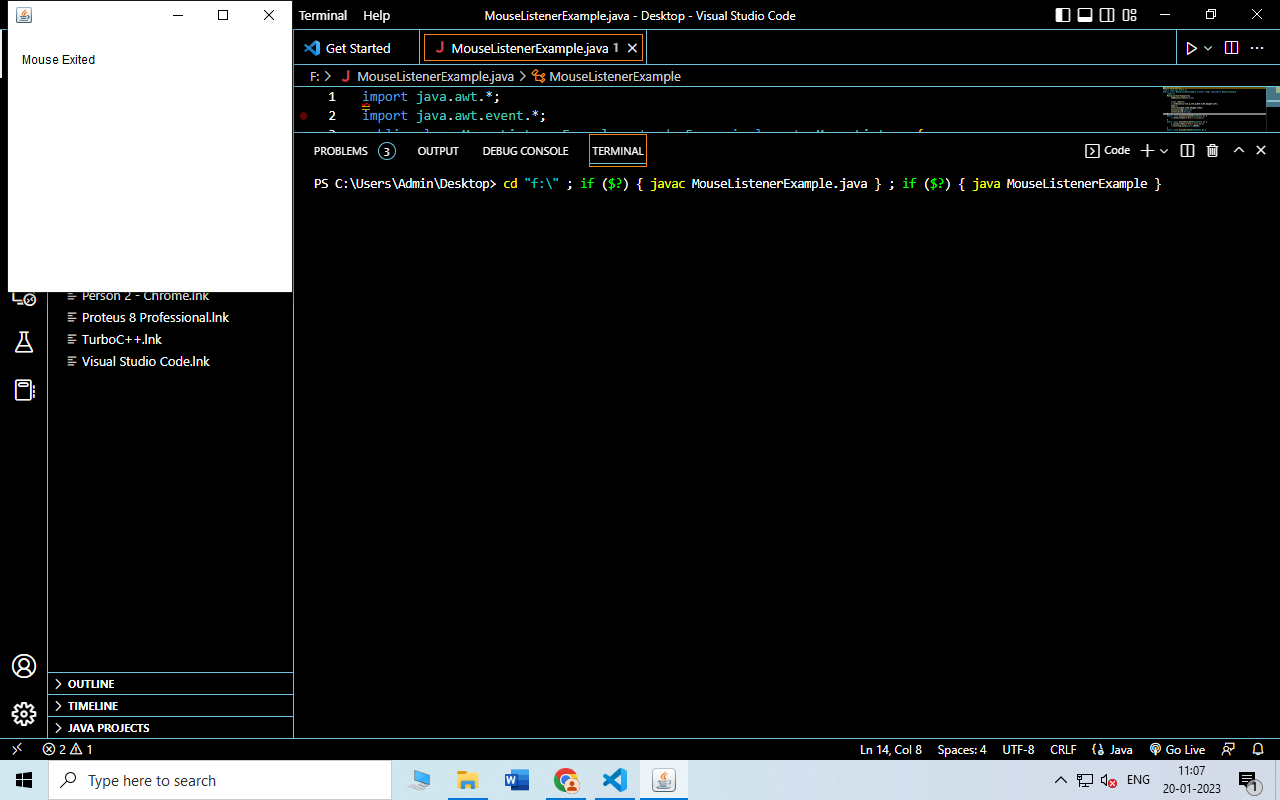
| import java.awt.\*;  import java.awt.event.\*;  public class MouseListenerExample extends Frame implements MouseListener{  Label l;  MouseListenerExample(){  addMouseListener(this);    l=new Label();  l.setBounds(20,50,100,20);  add(l);  setSize(300,300);  setLayout(null);  setVisible(true);  }  public void mouseClicked(MouseEvent e) {  l.setText("Mouse Clicked");  setBackground(Color.YELLOW);  }  public void mouseEntered(MouseEvent e) {  l.setText("Mouse Entered");  l.setForeground(Color.GREEN);  }  public void mouseExited(MouseEvent e) {  l.setText("Mouse Exited");  l.setForeground(Color.BLACK);  setBackground(Color.WHITE);  }  public void mousePressed(MouseEvent e) {  l.setText("Mouse Pressed");  }  public void mouseReleased(MouseEvent e) {  l.setText("Mouse Released");  }  public static void main(String[] args) {  new MouseListenerExample();  }  } |
| --- |

Output:

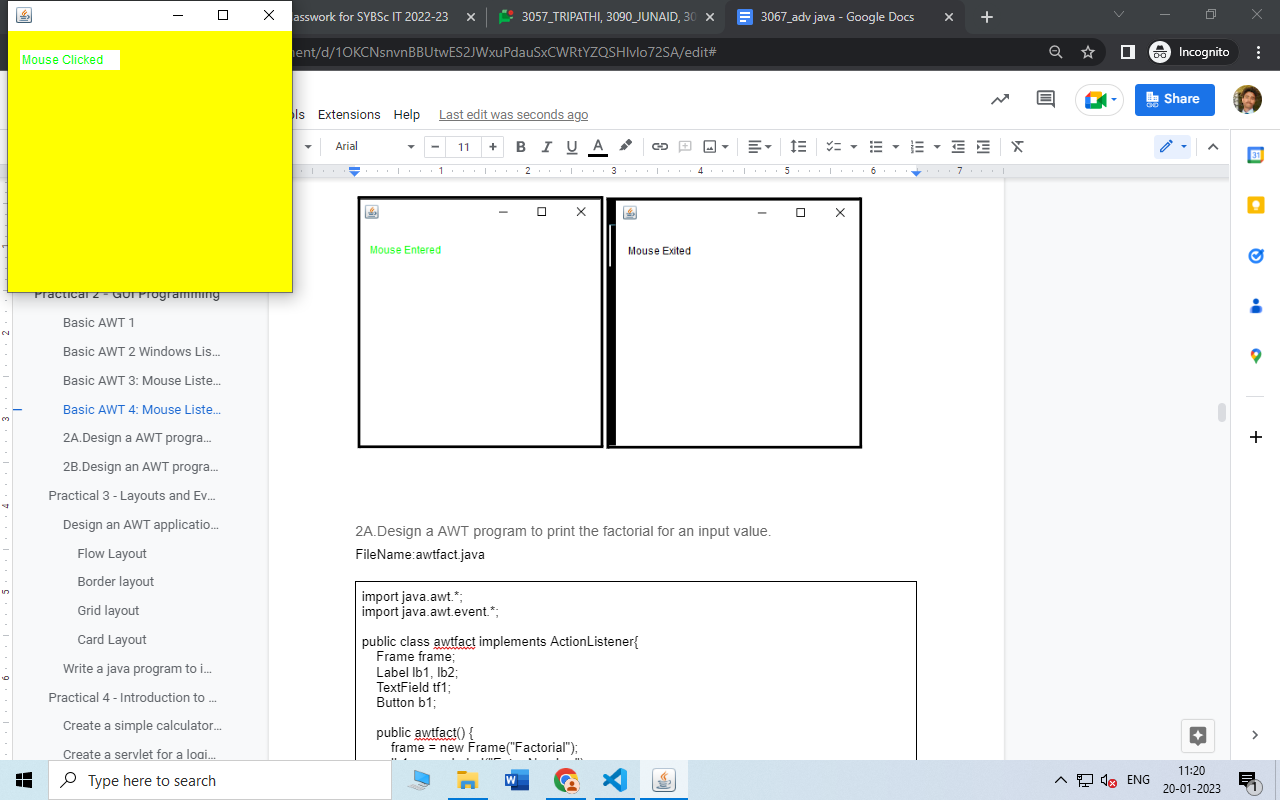
Label text color change on Entering:



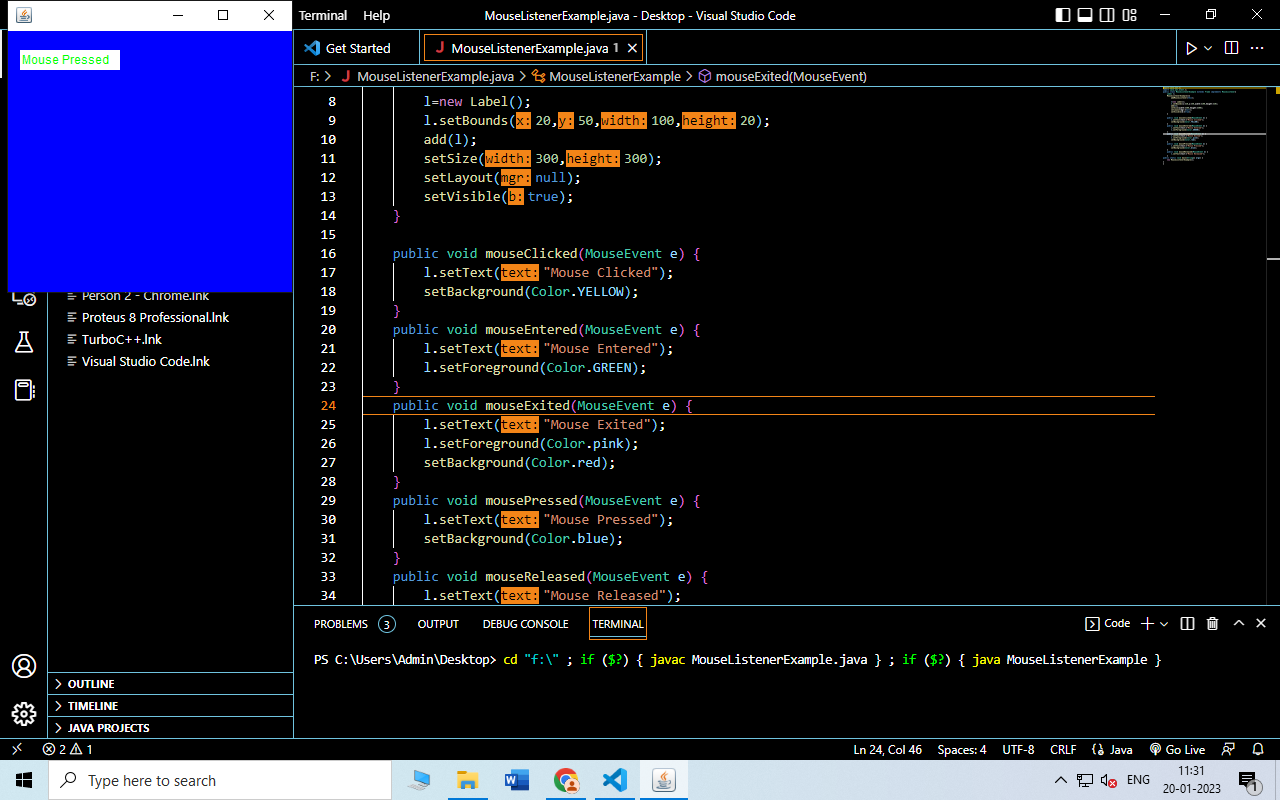
Color normal on exit:

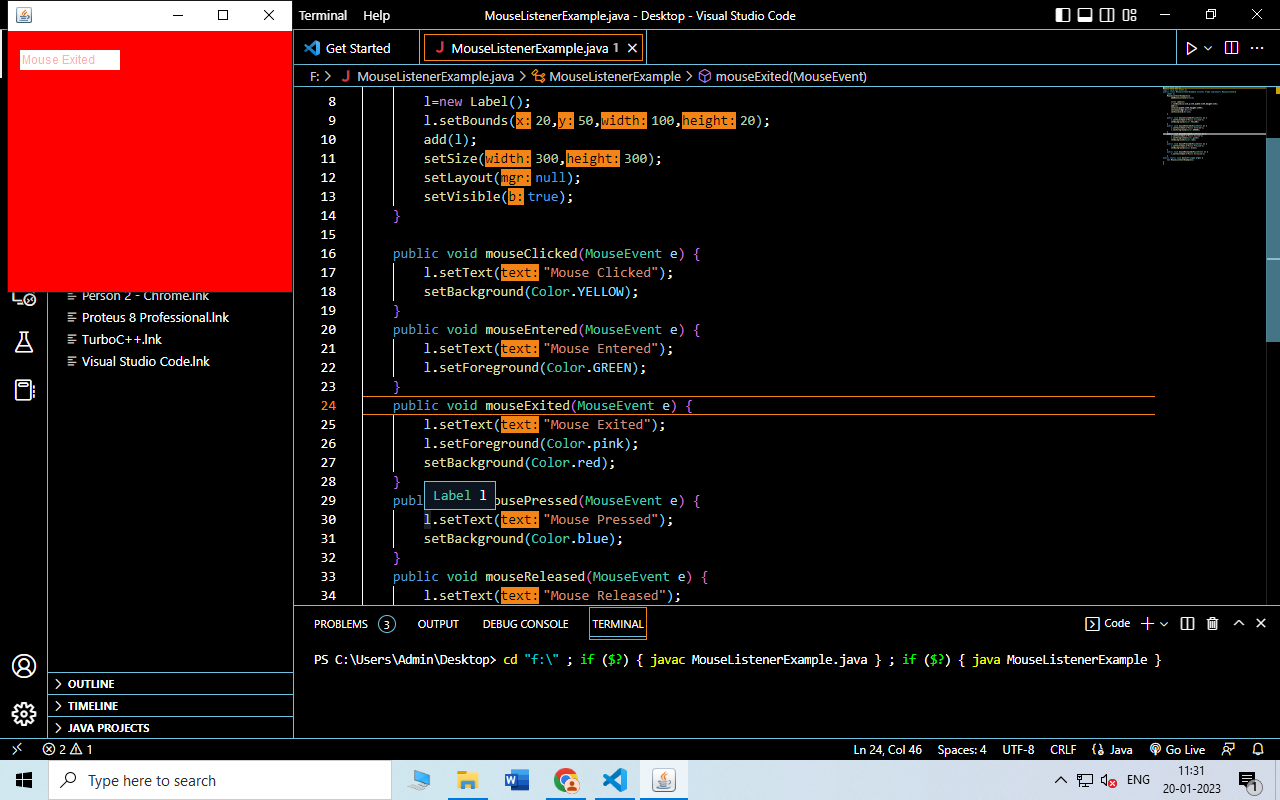


Color background on click:



Blue Background on press:



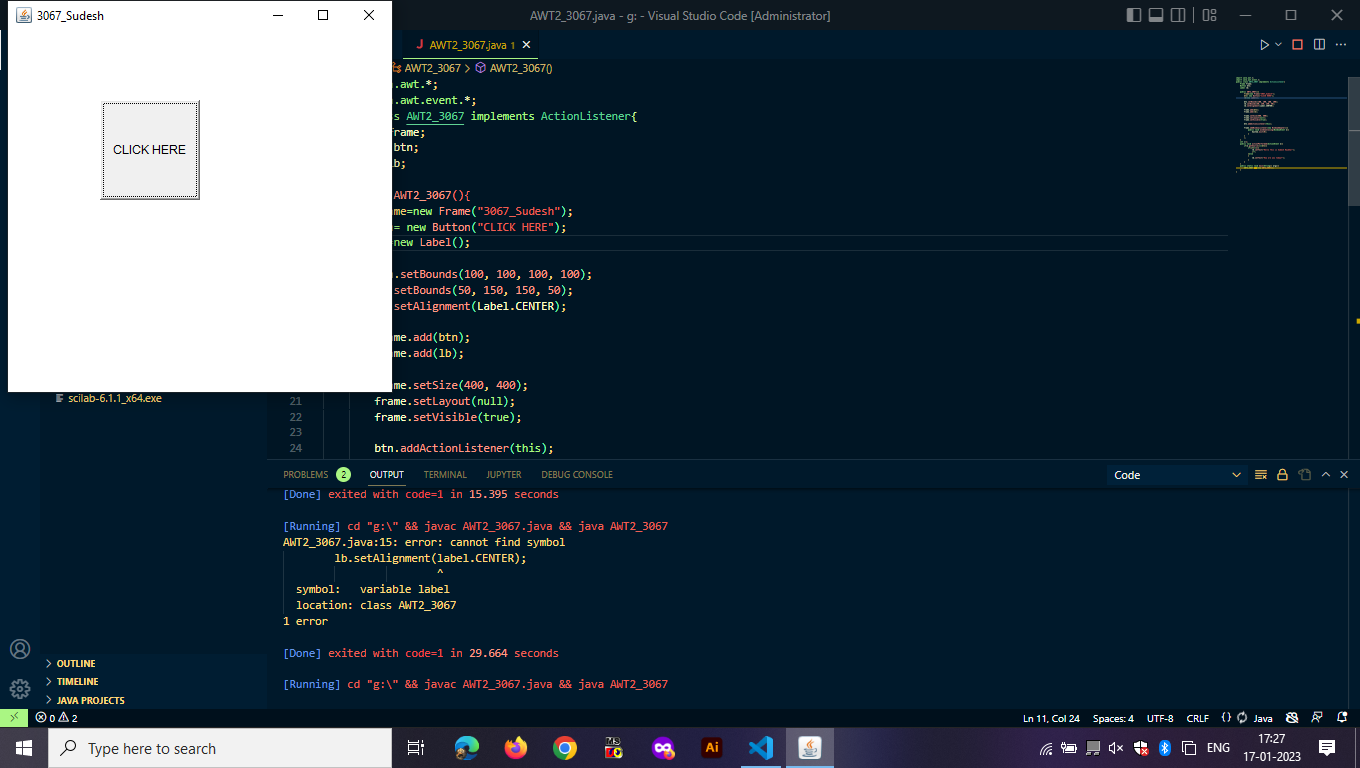


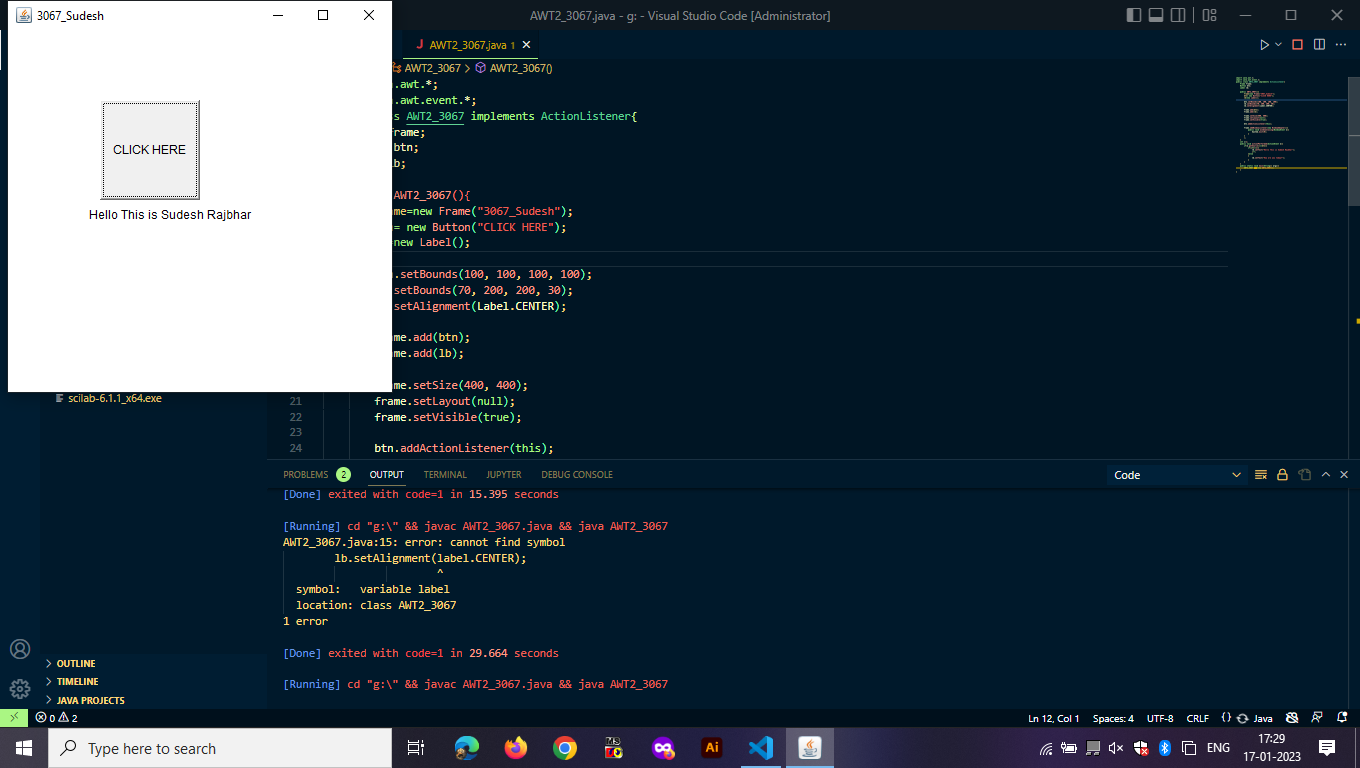
#### Windows Event

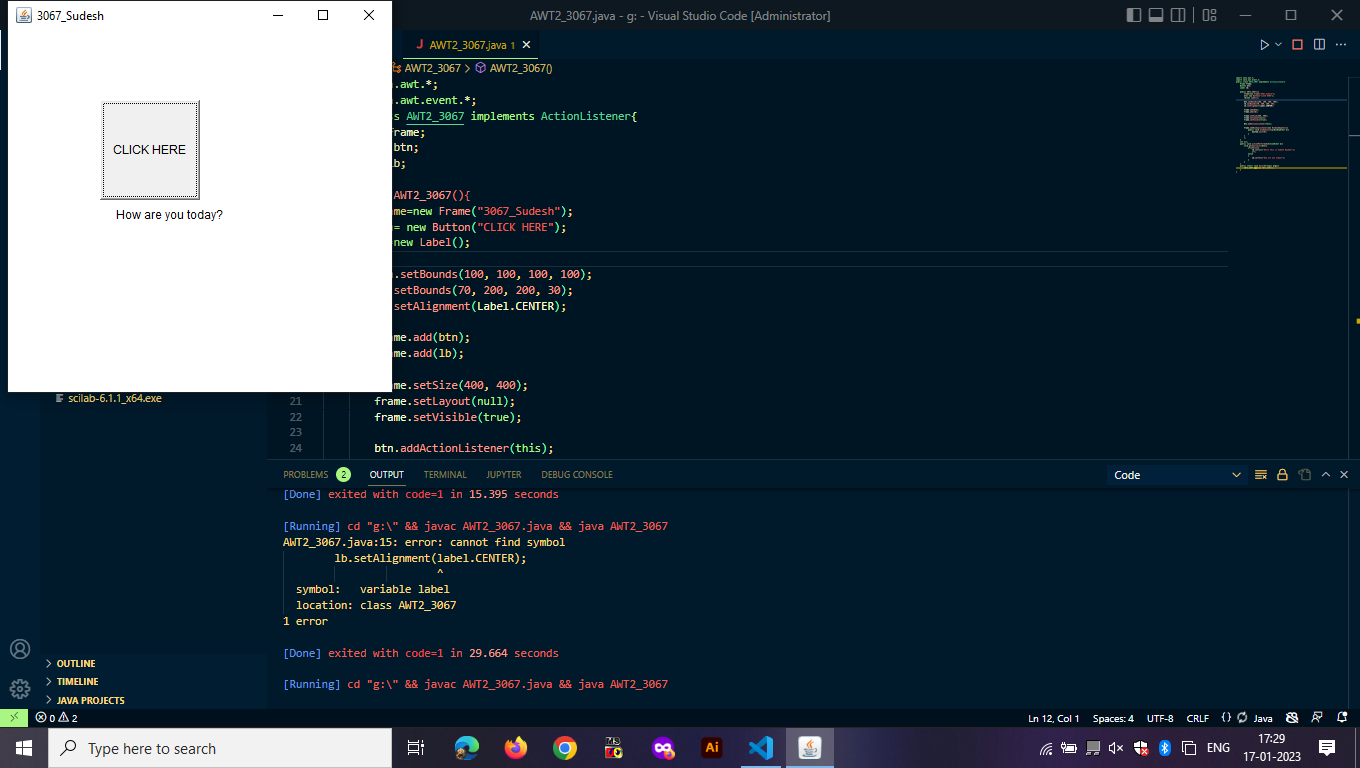
Filename: AWT2\_3067

| import java.awt.\*;  import java.awt.event.\*;  public class AWT2\_3067 implements ActionListener{  Frame frame;  Button btn;  Label lb;  public AWT2\_3067(){  frame=new Frame("3067\_Sudesh");  btn= new Button("CLICK HERE");  lb=new Label();  btn.setBounds(100, 100, 100, 100);  lb.setBounds(70, 200, 200, 30);  lb.setAlignment(Label.CENTER);  frame.add(btn);  frame.add(lb);  frame.setSize(400, 400);  frame.setLayout(null);  frame.setVisible(true);  btn.addActionListener(this);  frame.addWindowListener(new WindowAdapter(){  public void windowClosing(WindowEvent e){  System.exit(0);  }  }  );  }  int i=1;  public void actionPerformed(ActionEvent e){  if(e.getSource()==btn){  if(i%2==1){  lb.setText("Hello This is Sudesh Rajbhar");  i++;  }else  {  lb.setText("How are you today?");  }  }  }  public static void main(String[] args){  AWT2\_3067 awt=new AWT2\_3067();  }  } |
| --- |

Output:





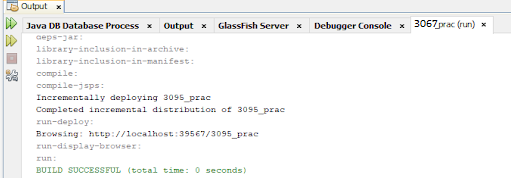


### 

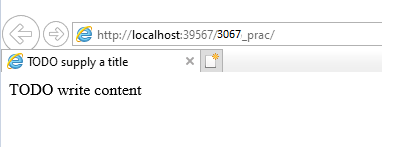
## Practical 4 - Introduction to Servlet Programming

#### 4A)TO DO Page:





Output:



#### 4B)Create s simple hello world program:

Code:

| /\*  \* To change this license header, choose License Headers in Project Properties.  \* To change this template file, choose Tools | Templates  \* and open the template in the editor.  \*/  package HelloWorld;  import java.io.IOException;  import java.io.PrintWriter;  import javax.servlet.ServletException;  import javax.servlet.annotation.WebServlet;  import javax.servlet.http.HttpServlet;  import javax.servlet.http.HttpServletRequest;  import javax.servlet.http.HttpServletResponse;  /\*\*  \*  \* @author Admin  \*/  @WebServlet(name = "HelloWorld", urlPatterns = {"/HelloWorld"})  public class HelloWorld extends HttpServlet {  /\*\*  \* Processes requests for both HTTP <code>GET</code> and <code>POST</code>  \* methods.  \*  \* @param request servlet request  \* @param response servlet response  \* @throws ServletException if a servlet-specific error occurs  \* @throws IOException if an I/O error occurs  \*/  protected void processRequest(HttpServletRequest request, HttpServletResponse response)  throws ServletException, IOException {  response.setContentType("text/html;charset=UTF-8");  try (PrintWriter out = response.getWriter()) {  /\* TODO output your page here. You may use following sample code. \*/  out.println("<!DOCTYPE html>");  out.println("<html>");  out.println("<head>");  out.println("<title>Servlet HelloWorld</title>");  out.println("</head>");  out.println("<body>");  out.println("<h1>Servlet HelloWorld at " + request.getContextPath() + "</h1>");  out.println("</body>");  out.println("</html>");  }  }  // <editor-fold defaultstate="collapsed" desc="HttpServlet methods. Click on the + sign on the left to edit the code.">  /\*\*  \* Handles the HTTP <code>GET</code> method.  \*  \* @param request servlet request  \* @param response servlet response  \* @throws ServletException if a servlet-specific error occurs  \* @throws IOException if an I/O error occurs  \*/  @Override  protected void doGet(HttpServletRequest request, HttpServletResponse response)  throws ServletException, IOException {  processRequest(request, response);  }  /\*\*  \* Handles the HTTP <code>POST</code> method.  \*  \* @param request servlet request  \* @param response servlet response  \* @throws ServletException if a servlet-specific error occurs  \* @throws IOException if an I/O error occurs  \*/  @Override  protected void doPost(HttpServletRequest request, HttpServletResponse response)  throws ServletException, IOException {  processRequest(request, response);  }  /\*\*  \* Returns a short description of the servlet.  \*  \* @return a String containing servlet description  \*/  @Override  public String getServletInfo() {  return "Short description";  }// </editor-fold>  } |
| --- |

Output:







c)Check even odd number:

Code:

Index.html:

| <!DOCTYPE html>  <!--  To change this license header, choose License Headers in Project Properties.  To change this template file, choose Tools | Templates  and open the template in the editor.  -->  <html>  <head>  <title>Calculator</title>  <meta charset="UTF-8">  <meta name="viewport" content="width=device-width, initial-scale=1.0">  </head>  <body>  <form method="get" action="simple\_c">  <h1>Calculator</h1>  First Number :  <input type="text" name="txtfno"/><br/>  Second Number :  <input type="text" name="txtsno"/><br/>  Select the Operation:<br/>    <input type="submit" name="operation" value="Addition">  <input type="submit" name="operation" value="Substraction">  <input type="submit" name="operation" value="Multiplication"><br>  <input type="submit" name="operation" value="Division">  <input type="submit" name="operation" value="Modulus">  <br/><br/>    <input type="reset" value="Reset"/>    </form>  </body>  </html> |
| --- |

helloworld.java:

| /\*  \* To change this license header, choose License Headers in Project Properties.  \* To change this template file, choose Tools | Templates  \* and open the template in the editor.  \*/  import java.io.IOException;  import java.io.PrintWriter;  import javax.servlet.ServletException;  import javax.servlet.annotation.WebServlet;  import javax.servlet.http.HttpServlet;  import javax.servlet.http.HttpServletRequest;  import javax.servlet.http.HttpServletResponse;  /\*\*  \*  \* @author Admin  \*/  @WebServlet(urlPatterns = {"/helloworld"})  public class helloworld extends HttpServlet {  /\*\*  \* Processes requests for both HTTP <code>GET</code> and <code>POST</code>  \* methods.  \*  \* @param request servlet request  \* @param response servlet response  \* @throws ServletException if a servlet-specific error occurs  \* @throws IOException if an I/O error occurs  \*/  protected void processRequest(HttpServletRequest request, HttpServletResponse response)  throws ServletException, IOException {  response.setContentType("text/html;charset=UTF-8")    try (PrintWriter out = response.getWriter()) {  /\* TODO output your page here. You may use following sample code. \*/  out.println("<!DOCTYPE html>");  out.println("<html>");  out.println("<head>");  out.println("<title>Servlet helloworld</title>");  out.println("</head>");  out.println("<body>");  String num=request.getParameter("number");  int number=Integer.parseInt(num);  out.println("<h1>Your number is: " + request.getParameter("number") + "</h1>");  if(number%2==0)  out.println("<h1> Number is even"+ "</h1>");  else  out.println("<h1> Number is odd"+ "</h1>");  out.println("</body>");  out.println("</html>");  }  }  // <editor-fold defaultstate="collapsed" desc="HttpServlet methods. Click on the + sign on the left to edit the code.">  /\*\*  \* Handles the HTTP <code>GET</code> method.  \*  \* @param request servlet request  \* @param response servlet response  \* @throws ServletException if a servlet-specific error occurs  \* @throws IOException if an I/O error occurs  \*/  @Override  protected void doGet(HttpServletRequest request, HttpServletResponse response)  throws ServletException, IOException {  processRequest(request, response);  }  /\*\*  \* Handles the HTTP <code>POST</code> method.  \*  \* @param request servlet request  \* @param response servlet response  \* @throws ServletException if a servlet-specific error occurs  \* @throws IOException if an I/O error occurs  \*/  @Override  protected void doPost(HttpServletRequest request, HttpServletResponse response)  throws ServletException, IOException {  processRequest(request, response);  }  /\*\*  \* Returns a short description of the servlet.  \*  \* @return a String containing servlet description  \*/  @Override  public String getServletInfo() {  return "Short description";  }// </editor-fold>  } |
| --- |

Output:



#### 4C)Create a simple calculator application using servlet.

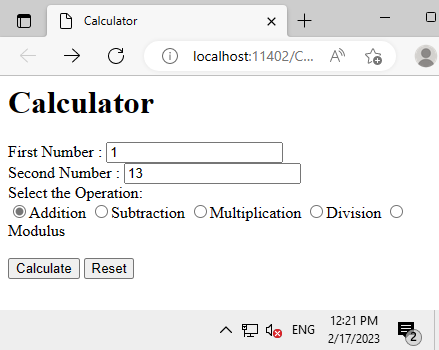
Code:

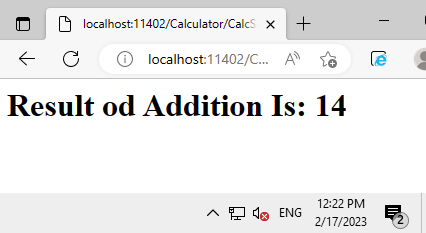
| <html>  <head>  <title>Calculator</title>  <meta charset="UTF-8">  <meta name="viewport" content="width=device-width, initial-scale=1.0">  </head>  <body>  <form method="get" action="CalcServ">  <h1>Calculator</h1>  First Number :  <input type="text" name="txtfno"/><br/>  Second Number :  <input type="text" name="txtsno"/><br/>  Select the Operation:<br/>    <input type="radio" name="operation" value="add">Addition  <input type="radio" name="operation" value="sub">Substraction  <input type="radio" name="operation" value="mul">Multiplication  <input type="radio" name="operation" value="div">Division  <input type="radio" name="operation" value="mod">Modulus  <br/><br/>    <input type="submit" value="Calculate"/>  <input type="reset" value="Reset"/>    </form>  </body>  </html> |
| --- |

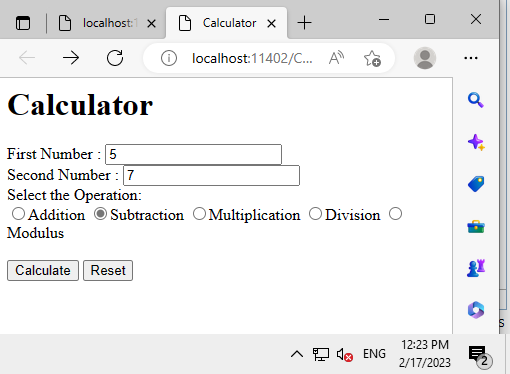
CalcServ.java:

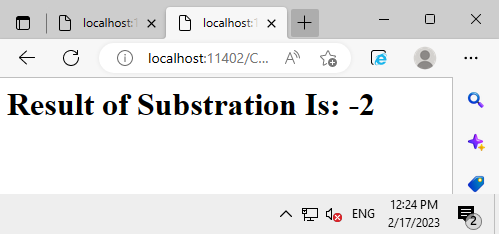
| package servlet;  import java.io.IOException;  import java.io.PrintWriter;  import javax.servlet.ServletException;  import javax.servlet.http.HttpServlet;  import javax.servlet.http.HttpServletRequest;  import javax.servlet.http.HttpServletResponse;  public class CalcServ extends HttpServlet {  protected void doGet(HttpServletRequest request, HttpServletResponse response)  throws ServletException, IOException  {  PrintWriter out = response.getWriter();  int x,y;  String str="";  x=Integer.parseInt(request.getParameter("txtfno"));  y=Integer.parseInt(request.getParameter("txtsno"));  str=request.getParameter("operation");  if(str.equals("add"))  out.println("<h1> Result od Addition Is: "+(x+y)+"</h1>");  else if(str.equals("sub"))  out.println("<h1> Result od Substration Is: "+(x-y)+"</h1>");  else if(str.equals("mul"))  out.println("<h1> Result od Multiplication Is: "+(x\*y)+"</h1>");  else if(str.equals("div"))  out.println("<h1> Result od Division Is: "+(x/y)+"</h1>");  else  out.println("<h1> Result od Modulus Is: "+(x%y)+"</h1>");  }  } |
| --- |

Output:









#### 4D)Create a servlet for a login page. If the username and password are correct then it says message “Hello <username>” else a message “login failed”.

Code:

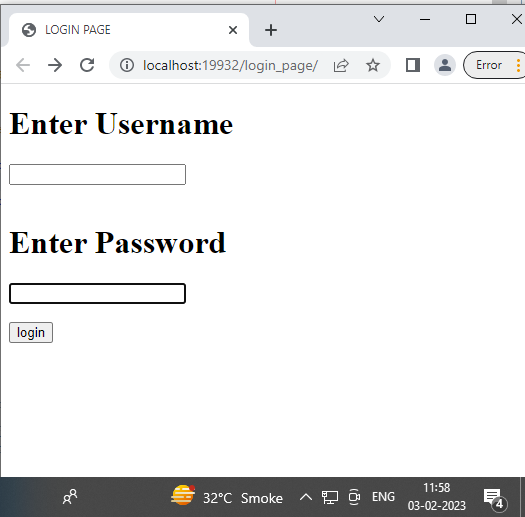
Index.html

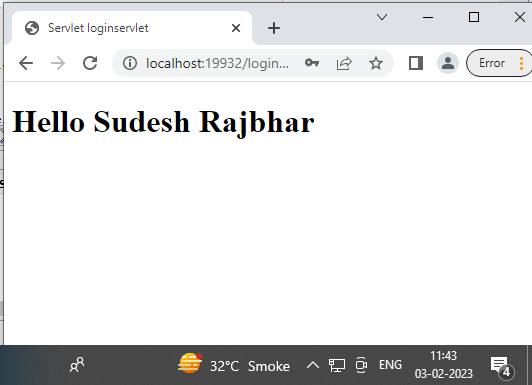
| <!DOCTYPE html>  <!--  To change this license header, choose License Headers in Project Properties.  To change this template file, choose Tools | Templates  and open the template in the editor.  -->  <html>  <head>  <title>LOGIN PAGE</title>  <meta charset="UTF-8">  <meta name="viewport" content="width=device-width, initial-scale=1.0">  </head>  <body>  <form action="loginservlet" method="get">  <input type="text" name="userName"/><br/><br/>  <input type="password" name="pass"/><br/><br/>  <input type="submit" value="login"/>  </form>  </body>  </html> |
| --- |

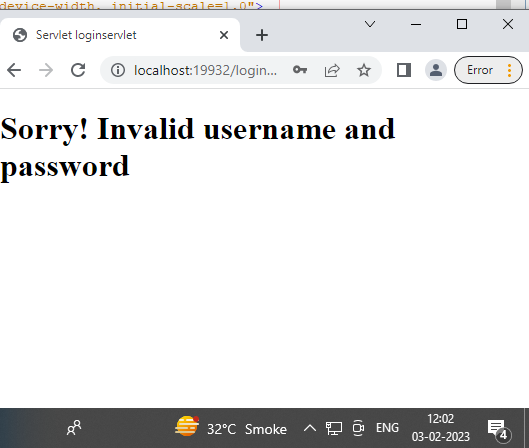
Login.java

| /\*  \* To change this license header, choose License Headers in Project Properties.  \* To change this template file, choose Tools | Templates  \* and open the template in the editor.  \*/  package loginservlet.java;  import java.io.IOException;  import java.io.PrintWriter;  import javax.servlet.ServletException;  import javax.servlet.http.HttpServlet;  import javax.servlet.http.HttpServletRequest;  import javax.servlet.http.HttpServletResponse;  public class loginservlet extends HttpServlet {  protected void processRequest(HttpServletRequest request, HttpServletResponse response)  throws ServletException, IOException {  response.setContentType("text/html;charset=UTF-8");  try (PrintWriter out = response.getWriter()) {    String userName=request.getParameter("userName");  String password=request.getParameter("pass");    String userName="Sudesh Rajbhar";  String pass="Sudesh@123";    if(userName.equals(userName) && pass.equals(pass)){  out.println("<!DOCTYPE html>");  out.println("<html>");  out.println("<head>");  out.println("<title>Servlet loginservlet</title>");  out.println("</head>");  out.println("<body>");  out.println("<h1>Hello "+ userName +"</h1>");  out.println("</body>");  out.println("</html>");  }  else{  out.println("<!DOCTYPE html>");  out.println("<html>");  out.println("<head>");  out.println("<title>Servlet loginservlet</title>");  out.println("</head>");  out.println("<body>");  out.println("<h1>Sorry! Invalid username and password</h1>");  out.println("</body>");  out.println("</html>");    }      }  }  @Override  protected void doGet(HttpServletRequest request, HttpServletResponse response)  throws ServletException, IOException {  processRequest(request, response);  }  @Override  protected void doPost(HttpServletRequest request, HttpServletResponse response)  throws ServletException, IOException {  processRequest(request, response);  }  @Override  public String getServletInfo() {  return "Short description";  }// </editor-fold>  } |
| --- |

Output:







.

## Practical 5 - Servlet Programming with Cookies .

### 

#### 5A.Create a servlet that uses Cookies to store the number of times a user has visited a servlet.

Code:

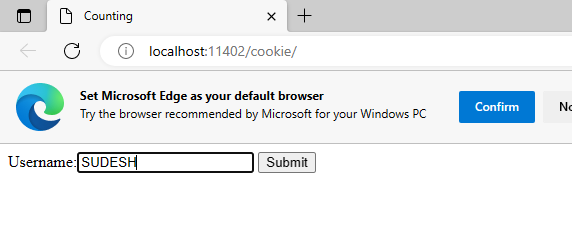
Index.html:

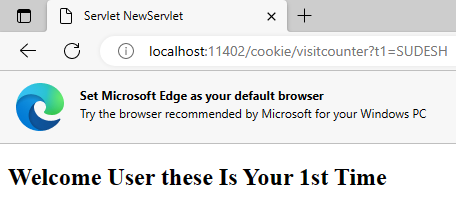
| <html>  <head>  <title>Counting</title>  <meta charset="UTF-8">  <meta name="viewport" content="width=device-width, initial-scale=1.0">  </head>  <body>  <form action="NewServlet" method="get">  Username:<input type="text" name="t1">  <input type="submit" >  </form>  </body>  </html> |
| --- |

visitcounter.java

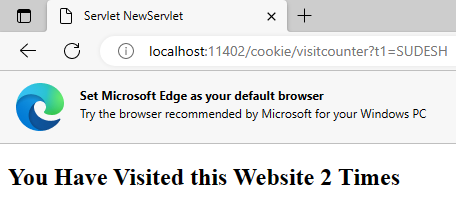
| import java.io.IOException;  import java.io.PrintWriter;  import javax.servlet.ServletException;  import javax.servlet.http.Cookie;  import javax.servlet.http.HttpServlet;  import javax.servlet.http.HttpServletRequest;  import javax.servlet.http.HttpServletResponse;  public class NewServlet extends HttpServlet {  static int i = 1;    protected void doGet(HttpServletRequest request, HttpServletResponse response)  throws ServletException, IOException {  response.setContentType("text/html;charset=UTF-8");  try ( PrintWriter out = response.getWriter()) {  out.println("<!DOCTYPE html>");  out.println("<html>");  out.println("<head>");  out.println("<title>Servlet NewServlet</title>");  out.println("</head>");  out.println("<body>");    Cookie c = new Cookie("visit",String.valueOf(i));  response.addCookie(c);    int j = Integer.parseInt(c.getValue());    if(j==1)  {  out.println("<h2>Welcome User these Is Your 1st Time</h2>");  }  else  {  out.println("<h2>You Have Visited this Website "+j+ " Times</h2>");  }  i++;    out.println("</body>");  out.println("</html>");  }  }  } |
| --- |

**When You visit for the first time:**





**When you visit for the second time:**



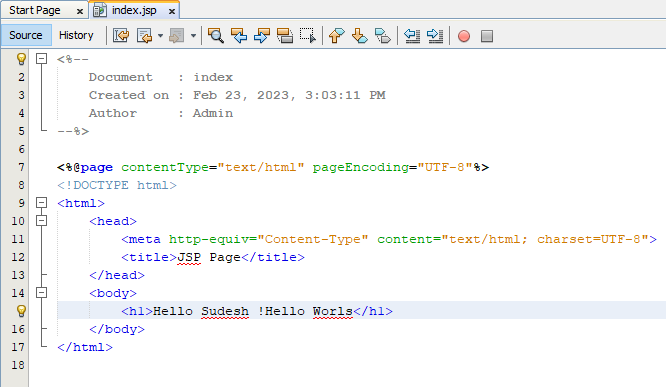
## 

## Practical 6 - JSP Programs

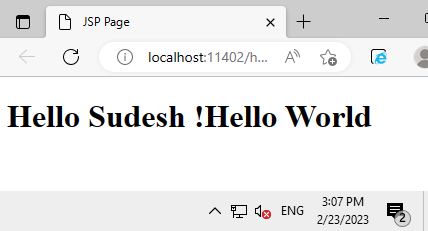
#### 6A.Simple JSP Application - Hello World.

Code:

| <%--  Document : index  Created on : Feb 23, 2023, 3:03:11 PM  Author : Admin  --%>  <%@page contentType="text/html" pageEncoding="UTF-8"%>  <!DOCTYPE html>  <html>  <head>  <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">  <title>JSP Page</title>  </head>  <body>  <h1>Hello World!</h1>  </body>  </html> |
| --- |



Output:



#### 6B. Using Request Dispatcher Interface create a Servlet which will validate the password entered by the user, if the user has entered "Servlet" as password, then he will be forwarded to Welcome Servlet else the user will stay on the index.html page and an error message will be displayed.

Index.html:

| <html>  <head>  <title>Login Page Information </title>  <meta charset="UTF-8">  <meta name="viewport" content="width=device-width, initial-scale=1.0">  </head>  <body>  <form action="Validate" method="get">  Name : <input type="text" name="user"> <br/>  Password : <input type="text" name="pass"> <br/>  <!-- comment -->  <input type="submit" value="SUBMIT">  </form>  </body>  </html> |
| --- |

Validate.java:

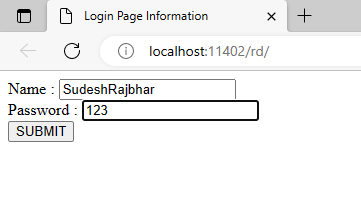
| package rd;  import java.io.IOException;  import java.io.PrintWriter;  import javax.servlet.RequestDispatcher;  import javax.servlet.ServletException;  import javax.servlet.http.HttpServlet;  import javax.servlet.http.HttpServletRequest;  import javax.servlet.http.HttpServletResponse;  public class Validate extends HttpServlet {  protected void doGet(HttpServletRequest request, HttpServletResponse response)  throws ServletException, IOException {  response.setContentType("text/html;charset=UTF-8");  try ( PrintWriter out = response.getWriter()) {  String name=request.getParameter("user");  String passwd=request.getParameter("pass");  if(passwd.equals("27062003"))  {  RequestDispatcher rd = request.getRequestDispatcher("Welcome");  rd.forward(request, response);  }  else  {  out.println("<font color='red'><b>You Have Entered Incorrect Password</b></font><br/>");  out.println("<a href= \"index.html\">LOGIN PAGE</a>");  }    }  }  } |
| --- |

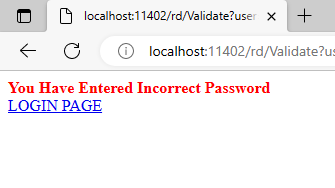
Welcome.java:

| package rd;  import java.io.IOException;  import java.io.PrintWriter;  import javax.servlet.ServletException;  import javax.servlet.http.HttpServlet;  import javax.servlet.http.HttpServletRequest;  import javax.servlet.http.HttpServletResponse;  public class Welcome extends HttpServlet {    protected void doGet(HttpServletRequest request, HttpServletResponse response)  throws ServletException, IOException {  response.setContentType("text/html;charset=UTF-8");  try ( PrintWriter out = response.getWriter())  { String NAME = request.getParameter("user");  out.println("<h1>Welcome " + NAME +"</h1" );  }  }  } |
| --- |

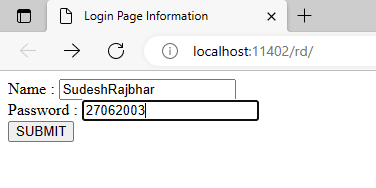
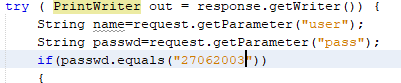
**Output**:

Wrong Password:





**Correct Password:**

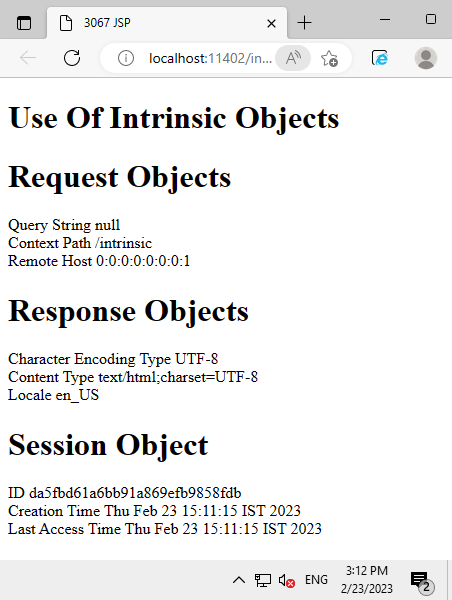


#### 6C.JSP using intrinsic objects - out, request, response, exception etc.JSP with validations

CODE:

| <%@page contentType="text/html" pageEncoding="UTF-8"%>  <!DOCTYPE html>  <html>  <head>  <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">  <title>3067 JSP</title>  </head>  <body>  <h1> Use Of Intrinsic Objects </h1>    <h1>Request Objects</h1>  Query String <%=request.getQueryString() %><br>  Context Path <%=request.getContextPath() %><br>  Remote Host <%=request.getRemoteHost() %><br>    <h1>Response Objects</h1>  Character Encoding Type <%=response.getCharacterEncoding() %><br>  Content Type <%=response.getContentType() %><br>  Locale <%=response.getLocale() %><br>    <h1>Session Object</h1>  ID <%=session.getId() %><br>  Creation Time <%=new java.util.Date(session.getCreationTime()) %><br>  Last Access Time <%=new java.util.Date(session.getLastAccessedTime()) %><br>  </body>  </html> |
| --- |

Output:



## Practical 7- Java Database Connectivity with JSP

### 

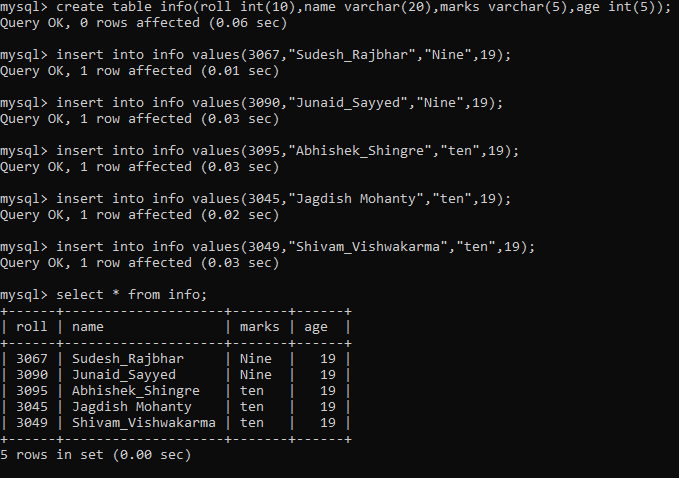
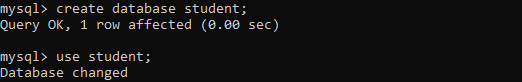
#### 7A)JSP to demonstrate database connectivity

Fetch student records

Insert records

### Fetch Records:

| Enter password: \*\*\*\*\*  Welcome to the MySQL monitor. Commands end with ; or \g.  Your MySQL connection id is 2  Server version: 5.1.28-rc-community MySQL Community Server (GPL)  Type 'help;' or '\h' for help. Type '\c' to clear the buffer.  mysql> create database student;  Query OK, 1 row affected (0.00 sec)  mysql> use student;  Database changed  mysql> create table info(roll int(10),name varchar(20),marks varchar(5),age int(5));  Query OK, 0 rows affected (0.06 sec)  mysql> insert into info values(3067,"Sudesh\_Rajbhar","Nine",19);  Query OK, 1 row affected (0.01 sec)  mysql> insert into info values(3090,"Junaid\_Sayyed","Nine",19);  Query OK, 1 row affected (0.03 sec)  mysql> insert into info values(3095,"Abhishek\_Shingre","ten",19);  Query OK, 1 row affected (0.03 sec)  mysql> insert into info values(3045,"Jagdish Mohanty","ten",19);  Query OK, 1 row affected (0.02 sec)  mysql> insert into info values(3049,"Shivam\_Vishwakarma","ten",19);  Query OK, 1 row affected (0.03 sec)  mysql> |
| --- |

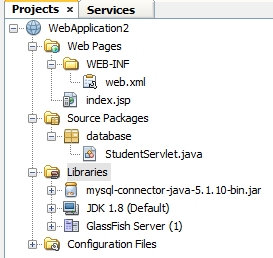


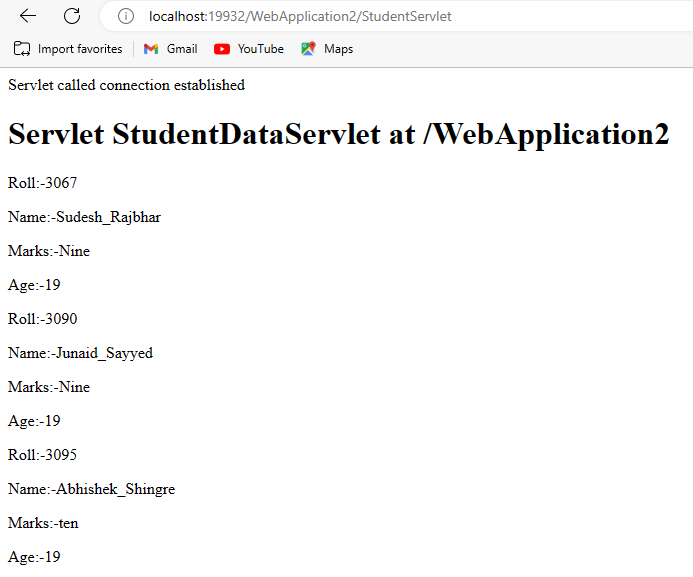
Index.jsp:

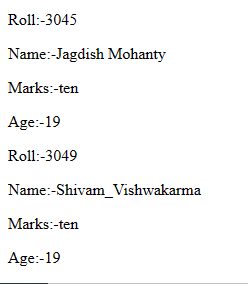
| <%--  Document : index  Created on : 14 Feb, 2023, 12:11:28 PM  Author : Admin  --%>  <%@page contentType="text/html" pageEncoding="UTF-8"%>  <!DOCTYPE html>  <html>  <head>  <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">  <title>JSP Page</title>  </head>  <body>  <h1>Student Data</h1>  <form action="StudentServlet" method="post">  <input type="submit" value="click to fetch"/>  </form>  </body>  </html> |
| --- |

StudentServlet.java:

| package database;  import java.io.IOException;  import java.io.PrintWriter;  import javax.servlet.ServletException;  import javax.servlet.http.HttpServlet;  import javax.servlet.http.HttpServletRequest;  import javax.servlet.http.HttpServletResponse;  import java.sql.\*;  public class StudentServlet extends HttpServlet {  protected void processRequest(HttpServletRequest request, HttpServletResponse response)  throws ServletException, IOException {  response.setContentType("text/html;charset=UTF-8");  try (PrintWriter out = response.getWriter()) {  /\* TODO output your page here. You may use following sample code. \*/  out.println("Servlet called");  Class.forName("com.mysql.jdbc.Driver");  Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3308/student","root","mysql");  out.println("connection established");  Statement st = con.createStatement();  ResultSet rs= st.executeQuery("select \* from info");    out.println("<!DOCTYPE html>");  out.println("<html>");  out.println("<head>");  out.println("<title>Servlet StudentDataServlet</title>");  out.println("</head>");  out.println("<body>");  out.println("<h1>Servlet StudentDataServlet at " + request.getContextPath() + "</h1>");  while (rs.next())  {  out.println("<p> Roll:-" + rs.getInt(1) + "</p>");  out.println("<p> Name:-" + rs.getString(2)+ "</p>");  out.println("<p> Marks:-" + rs.getString(3)+"</p>");  out.println("<p> Age:-" + rs.getInt(4) + "</p>");    }    out.println("</body>");  out.println("</html>");  }  catch(Exception e){System.out.println(e);}  }  // <editor-fold defaultstate="collapsed" desc="HttpServlet methods. Click on the + sign on the left to edit the code.">  /\*\*  \* Handles the HTTP <code>GET</code> method.  \*  \* @param request servlet request  \* @param response servlet response  \* @throws ServletException if a servlet-specific error occurs  \* @throws IOException if an I/O error occurs  \*/  @Override  protected void doGet(HttpServletRequest request, HttpServletResponse response)  throws ServletException, IOException {  processRequest(request, response);  }  /\*\*  \* Handles the HTTP <code>POST</code> method.  \*  \* @param request servlet request  \* @param response servlet response  \* @throws ServletException if a servlet-specific error occurs  \* @throws IOException if an I/O error occurs  \*/  @Override  protected void doPost(HttpServletRequest request, HttpServletResponse response)  throws ServletException, IOException {  processRequest(request, response);  }  /\*\*  \* Returns a short description of the servlet.  \*  \* @return a String containing servlet description  \*/  @Override  public String getServletInfo() {  return "Short description";  }// </editor-fold>} |
| --- |







### Insert records:

| **Problem Definition**  Create a Java EE 6 web application to connect to a database in MYSQL. Create a db, RJCIT, create a table student with three columns - Roll, Name, Class. Add the necessary JAR file to the NETBEANS project. Create an index. jsp to accept the values "Roll, Name and Class". the page should have a submit button, in response, a servlet should be called. This servlet will display a message showing that the data was successfully entered. Add a button to the servlet to navigate to a second servlet, 'ShowStudent'. This servlet should display the data from the student table in tabular format, including the last entered value. Expected outputs are attached herewith. |
| --- |

**Database Creation:**

| Enter password: \*\*\*\*\*  Welcome to the MySQL monitor. Commands end with ; or \g.  Your MySQL connection id is 1  Server version: 5.1.28-rc-community MySQL Community Server (GPL)  Type 'help;' or '\h' for help. Type '\c' to clear the buffer.  mysql> create database rjcit;  Query OK, 1 row affected (0.00 sec)  mysql> use rjcit;  Database changed  mysql> create table student(Roll int(5), Name varchar(20),Class varchar(10));  Query OK, 0 rows affected (0.12 sec)  mysql> insert into student values(3067,"Sudesh\_Rajbhar","SYIT");  Query OK, 1 row affected (0.09 sec)  mysql> insert into student values(3057,"Raj Tripathi","SYIT");  Query OK, 1 row affected (0.07 sec)  mysql> insert into student values(3095,"Abhishek\_Shingre","SYIT");  Query OK, 1 row affected (0.03 sec)  mysql> insert into student values(3090,"Junaid\_Sayed","SYIT");  Query OK, 1 row affected (0.02 sec)  mysql> insert into student values(3049,"Shivam Vishwakarma","SYIT");  Query OK, 1 row affected (0.02 sec)  mysql> insert into student values(3045,"Jagadish Mohanty","SYIT");  Query OK, 1 row affected (0.08 sec)  mysql> insert into student values(3021,"Vivek Yadav","SYIT");  Query OK, 1 row affected (0.03 sec)  mysql> insert into student values(3020,"Farhan Shaikh","SYIT");  Query OK, 1 row affected (0.08 sec)  mysql> insert into student values(3056,"Omkar Shinde","SYIT");  Query OK, 1 row affected (0.09 sec)  mysql> insert into student values(3001,"Sahil Naik","FYIT");  Query OK, 1 row affected (0.12 sec)  mysql> |
| --- |

| **NEWLY CREATED TABLE** |
| --- |

Index.jsp:

| <body>  <div>Database Connectivity</div>  <form action="StudentDataServlet" method="post">  Student Roll No :<input type="text" name ="txtRoll" /><br>  Student Name : <input type="text" name="txtName"/><br>  Class : <input type ="text" name="txtClass"/><br>  <input type="submit" name="submit" value="Click to Insert"/>  </form>  </body> |
| --- |

StudentDataServlet.java

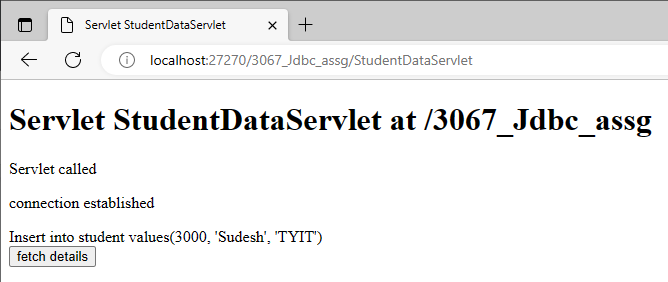
| package database;  import java.sql.\*;  import com.mysql.jdbc.Connection;  import java.io.IOException;  import java.io.PrintWriter;  import java.sql.DriverManager;  import java.sql.SQLException;  import java.util.logging.Level;  import java.util.logging.Logger;  import javax.servlet.ServletException;  import javax.servlet.http.HttpServlet;  import javax.servlet.http.HttpServletRequest;  import javax.servlet.http.HttpServletResponse;  public class StudentDataServlet extends HttpServlet {  protected void processRequest(HttpServletRequest request, HttpServletResponse response)  throws ServletException, IOException, SQLException, ClassNotFoundException {  response.setContentType("text/html;charset=UTF-8");  try (PrintWriter out = response.getWriter()) {  /\* TODO output your page here. You may use following sample code. \*/  out.println("<!DOCTYPE html>");  out.println("<html>");  out.println("<head>");  out.println("<title>Servlet StudentDataServlet</title>");  out.println("</head>");  out.println("<body>");  out.println("<h1>Servlet StudentDataServlet at " + request.getContextPath() + "</h1>");    out.println("<p>Servlet called</p>");  Class.forName("com.mysql.jdbc.Driver");  Connection con = (Connection) DriverManager.getConnection("jdbc:mysql://localhost:3307/rjcit","root","mysql");  out.println("<p>connection established</p>");  Statement st = (Statement)con.createStatement();    int rollno = Integer.parseInt(request.getParameter("txtRoll"));  String studentName = request.getParameter("txtName");  String studentClass = request.getParameter("txtClass");    String sqlQuery = "Insert into student values(" + rollno + ", '" + studentName + "', '" + studentClass + "')";  out.println(sqlQuery);  st.executeUpdate(sqlQuery);    out.println("<form action ='ShowStudent' method = 'post'>");  out.println("<input type = 'submit' value = 'fetch details'/>");  out.println("</form>");  out.println("</body>");  out.println("</html>");        }  }  @Override  protected void doGet(HttpServletRequest request, HttpServletResponse response)  throws ServletException, IOException {  try {  processRequest(request, response);  } catch (SQLException ex) {  Logger.getLogger(StudentDataServlet.class.getName()).log(Level.SEVERE, null, ex);  } catch (ClassNotFoundException ex) {  Logger.getLogger(StudentDataServlet.class.getName()).log(Level.SEVERE, null, ex);  }  }  @Override  protected void doPost(HttpServletRequest request, HttpServletResponse response)  throws ServletException, IOException {  try {  processRequest(request, response);  } catch (SQLException ex) {  Logger.getLogger(StudentDataServlet.class.getName()).log(Level.SEVERE, null, ex);  } catch (ClassNotFoundException ex) {  Logger.getLogger(StudentDataServlet.class.getName()).log(Level.SEVERE, null, ex);  }  }    @Override  public String getServletInfo() {  return "Short description";  }  } |
| --- |

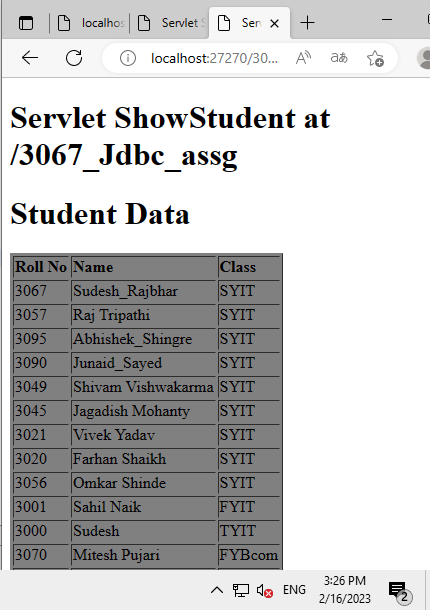
ShowStudent.java:

| package database;  import java.sql.\*;  import java.io.IOException;  import java.io.PrintWriter;  import java.sql.Connection;  import java.sql.DriverManager;  import java.sql.ResultSet;  import java.sql.SQLException;  import java.sql.Statement;  import java.util.logging.Level;  import java.util.logging.Logger;  import javax.servlet.ServletException;  import javax.servlet.http.HttpServlet;  import javax.servlet.http.HttpServletRequest;  import javax.servlet.http.HttpServletResponse;  public class ShowStudent extends HttpServlet {  protected void processRequest(HttpServletRequest request, HttpServletResponse response)  throws ServletException, IOException, ClassNotFoundException, SQLException {  response.setContentType("text/html;charset=UTF-8");  try (PrintWriter out = response.getWriter()) {    out.println("<!DOCTYPE html>");  out.println("<html>");  out.println("<head>");  out.println("<title>Servlet ShowStudent</title>");  out.println("</head>");  out.println("<body>");  out.println("<h1>Servlet ShowStudent at " + request.getContextPath() + "</h1>");  Class.forName("com.mysql.jdbc.Driver");  Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3307/rjcit", "root", "mysql");  Statement st1 = (Statement) con.createStatement();  String qry = "Select \* from student";  ResultSet rs1 = st1.executeQuery(qry);  out.println("<h1>Student Data</h1>");  out.println("<table border = '1' bgcolor = 'grey'>");  out.println("<tr>");  out.println("<td><b>Roll No</b></td>");  out.println("<td><b>Name</b></td>");  out.println("<td><b>Class</b></td>");  out.println("</tr>");  while (rs1.next()) {  out.println("<tr>");  out.println("<td>" + rs1.getInt("roll\_no") + "</td>");  out.println("<td>" + rs1.getString("name") + "</td>");  out.println("<td>" + rs1.getString("class") + "</td>");  out.println("</tr>");  }  out.println("</table>");  out.println("</body>");  out.println("</html>");  }  }  @Override  protected void doGet(HttpServletRequest request, HttpServletResponse response)  throws ServletException, IOException {  try {  processRequest(request, response);  } catch (ClassNotFoundException ex) {  Logger.getLogger(ShowStudent.class.getName()).log(Level.SEVERE, null, ex);  } catch (SQLException ex) {  Logger.getLogger(ShowStudent.class.getName()).log(Level.SEVERE, null, ex);  }  }  @Override  protected void doPost(HttpServletRequest request, HttpServletResponse response)  throws ServletException, IOException {  try {  processRequest(request, response);  } catch (ClassNotFoundException ex) {  Logger.getLogger(ShowStudent.class.getName()).log(Level.SEVERE, null, ex);  } catch (SQLException ex) {  Logger.getLogger(ShowStudent.class.getName()).log(Level.SEVERE, null, ex);  }  }  @Override  public String getServletInfo() {  return "Short description";  }// </editor-fold>  } |
| --- |

**Adding New Data :**





****

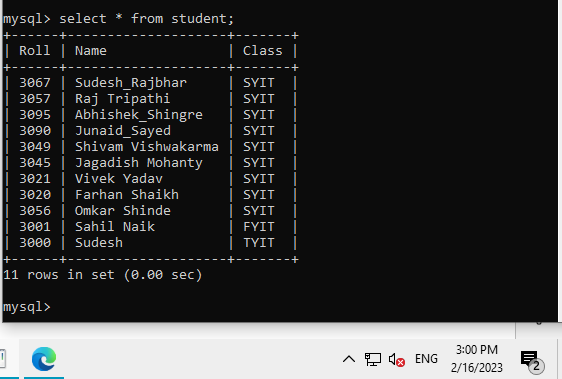
**NEW ROW ADDED WITH :**

**ROLL NO:3000**

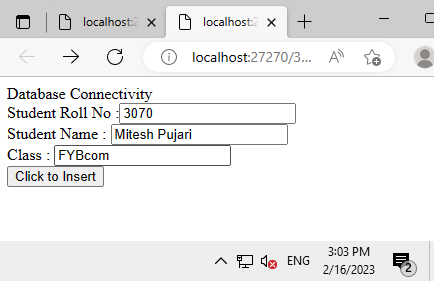
**NAME:SUDESH**

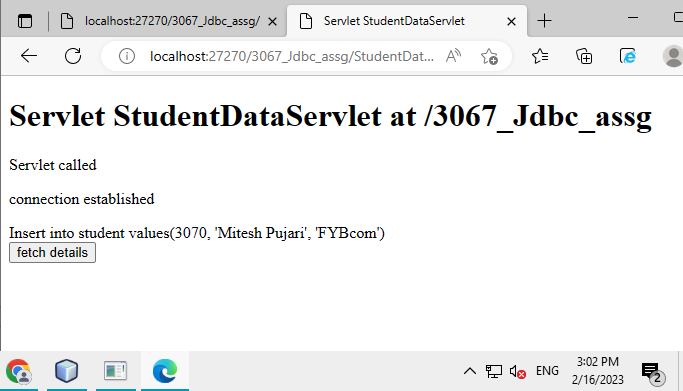
**CLASS:TYIT**

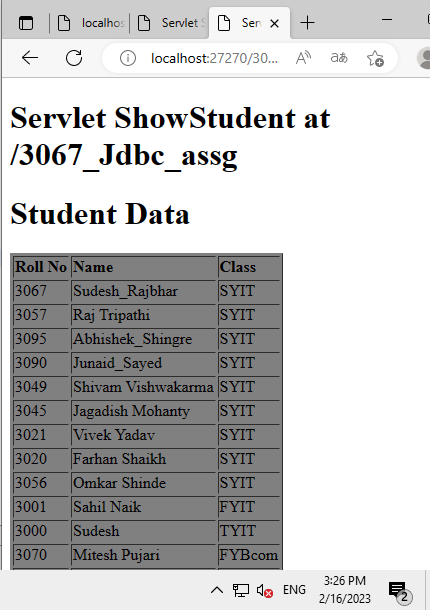
****

****

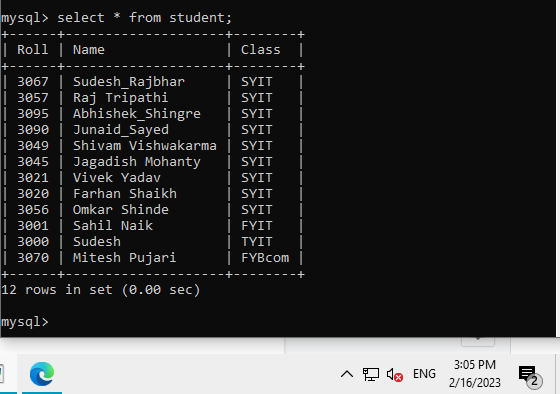
**One more Example:**

****

****

****

****

****

## Practical 8 -Networking in Java

### 

#### 8A)Create a socket program to implement TCP protocol

Code:

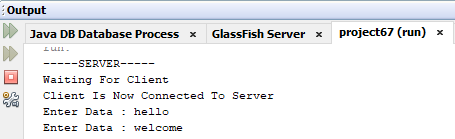
**MyServer.java:**

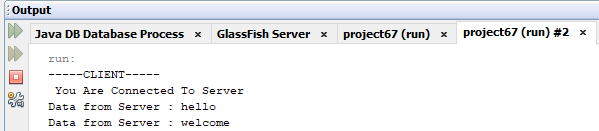
| package socket;  import java.net.\*;  import java.io.\*;  import java .util.\*;  public class MyServer  {  public static void main(String[] args)  {  try  {  ServerSocket ss = new ServerSocket(3045); // portno.  System.out.println("-----SERVER-----");  System.out.println("Waiting For Client");  Socket server = ss.accept();  System.out.println("Client Is Now Connected To Server");    // Send Data From TCPServer To Client  Scanner sc = new Scanner(System.in);  PrintWriter pw=new PrintWriter(server.getOutputStream());  while(true)  {  System.out.print("Enter Data : ");  String data=sc.nextLine();  pw.println("Data from Server : "+data);  pw.flush();  if(data.equals("exit"))  break;  }    }  catch(Exception ex)  {  System.out.println("Problem Occur In Connecting t o The Sever ");  }  }  } |
| --- |

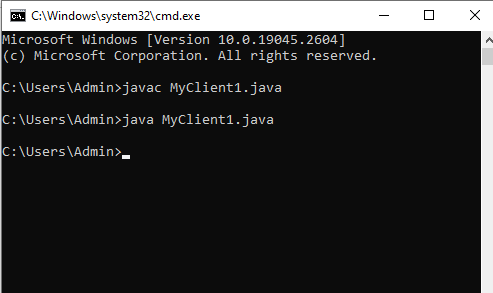
**MyClient**:

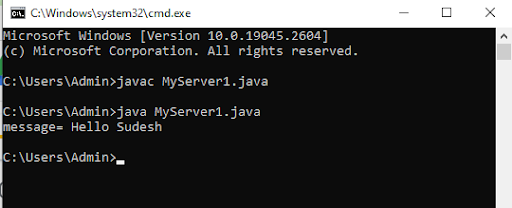
| package socket;  import java.net.\*;  import java.io.\*;  public class MyClient  {  public static void main(String[] args)  {  try  {  System.out.println("-----CLIENT-----");  Socket client = new Socket("localhost",3045);  System.out.println(" You Are Connected To Server");  //read Data From Server  BufferedReader br = new BufferedReader(new InputStreamReader(client.getInputStream()));  while (true)  {  String data=br.readLine();  System.out.println(data);  if(data.equals("exit"))  {  System.out.println("Connection Ended");  break;  }  }  client.close();  br.close();  }  catch(SocketException e)  {  System.out.println("Connection Ended ");  }  catch (Exception ex)  {  System.out.println("Error Occured On Connecting To Server ");  }  }    } |
| --- |

Output:









#### 8B)Write a java program to implement UDP protocol

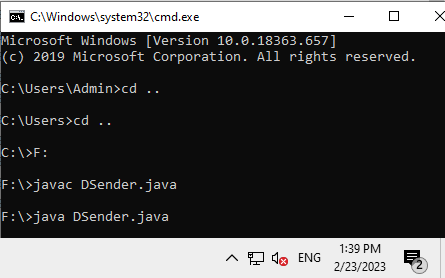
Codee:

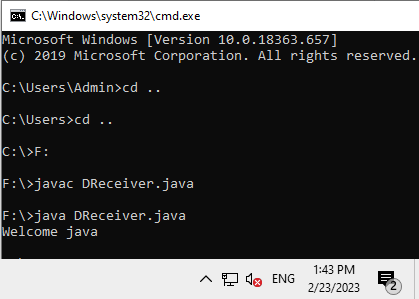
DSender.java:

| //DSender.java  import java.net.\*;  public class DSender{  public static void main(String[] args) throws Exception {  DatagramSocket ds = new DatagramSocket();  String str = "Welcome java";  InetAddress ip = InetAddress.getByName("127.0.0.1");    DatagramPacket dp = new DatagramPacket(str.getBytes(), str.length(), ip, 3000);  ds.send(dp);  ds.close();  }  } |
| --- |

DReceiver.java:

| //DReceiver.java  import java.net.\*;  public class DReceiver{  public static void main(String[] args) throws Exception {  DatagramSocket ds = new DatagramSocket(3000);  byte[] buf = new byte[1024];  DatagramPacket dp = new DatagramPacket(buf, 1024);  ds.receive(dp);  String str = new String(dp.getData(), 0, dp.getLength());  System.out.println(str);  ds.close();  }  } |
| --- |





## Practical 9 - Implement the following Hibernate applications.

### 

#### 9A)Develop an application to demonstrate Hibernate One - To -One Mapping Using Annotation.

