Java Lab

OBCA-DS-454

Manav Rachna International Institute of Research and Studies

School of Computer Applications

Department of Computer Applications

Submitted By	
Student Name	**Dhruv Joshi**
Roll No	**23/SCA/BCA(DS&BDA)/010**
Programme	**Bachelor of Computer Applications**
Semester	**4th Semester**
Section	**E**
Department	**Computer Applications**
Batch	**2023-26**
Submitted To	
Faculty Name	**Dr. Priyanka Sharma**

Manav Rachna International Institute of Research and Studies
Deemed to be University under section 3 of the UGC Act, 1956

SCHOOL OF COMPUTER APPLICATIONS

Sr. No.	Title	Date	Signature
1	Write a program to find the average and sum of the N numbers using Command line argument.		
2	Write a program to demonstrate type casting.		
3.	Write a program to generate prime numbers between 1 & given number		
4.	Write a program to design a class account using the inheritance and static members which show all functions of a bank (Withdrawl, deposit)		
5.	Write a program to create a simple class to find out the area and perimeter of rectangle using super and this keyword.		
6.	Write a program to find the factorial of a given number using recursion.		
7.	Write a program to design a class using abstract methods and abstract classes.		
8.	Write a program to count the number of objects created for a class using static member function		
9.	Write a program to demonstrate the use of function overloading.		
10.	Write a program to demonstrate the use of multiple inheritance.		
11.	Write a program that show the partial implementation of Interface		
12.	Write a program to design a string class that perform string method(Equal, Reverse the string, change case).		
13.	Write a program to handle the exception using try and multiple catch block.		

14.	Write a program to create a package that access the member of External class as well as same package.		
15.	Write a program that import the user define package and access the Member variable of classes that contained by package.		
16.	Write a program to handle the user defined exception using throw keyword.		
17.	Write a program to create a class component that shows controls and event handling on that controls.		
18.	(mathcalc).		
	Write a program to draw the line, Rectangle, oval, text using the graphics method.		
19.	Write a program to create a menu using the frame.		
20.	Write a program to create a menu using the frame.		
21.	Write a program to implement the flow layout and border layout.		
22.	Write a program to imp Write a program to create a dialogbox. lement the gridLayout, cardLayout.		
23.	Write a program to implement the gridLayout, cardLayout.		
24.	Write a program to create Frame that display the student information		

Experiment No: 1

Write a program to find the average and sum of the N numbers using Command line argument

```
 public class SumAverage {
public static void main(String[] args) {
if (args.length == 0) {
System.out.println("Please provide numbers as command-line arguments.");
return;
}
int sum = 0;
for (String arg : args) {
try {
int num = Integer.parseInt(arg);
sum += num;
} catch (NumberFormatException e) {
System.out.println(arg + " is not a valid integer.");
return;
}
}
double average = (double) sum / args.length;
System.out.println("Sum = " + sum);
System.out.println("Average = " + average);
}
}
```

Output:

```
Sum: 150.0
Average: 30.0
```

Experiment No: 2

Write a program to demonstrate type casting.

```
public class TypeCastingDemo {
public static void main(String[] args) {
// Implicit type casting (widening)
int intValue = 10;
double doubleValue = intValue; // int to double
System.out.println("Implicit Type Casting:");
System.out.println("Integer Value: " + intValue);
System.out.println("Double Value after implicit casting: " + doubleValue);
// Explicit type casting (narrowing)
double anotherDoubleValue = 7.78;
int anotherIntValue = (int) anotherDoubleValue; // double to int
System.out.println("\nExplicit Type Casting:");
System.out.println("Double Value: " + anotherDoubleValue);
System.out.println("Integer Value after explicit casting: " + anotherIntValue);
}
}
```

Result/Output:

```
Output

Implicit Type Casting:
Integer Value: 10
Double Value after implicit casting: 10.0

Explicit Type Casting:
Double Value: 7.78
Integer Value after explicit casting: 7
```

Experiment No: 3

Write a program to generate prime numbers between 1 & given number

```
 import java.util.Scanner;
public class PrimeNumberGenerator {
public static void main(String[] args) {
Scanner scanner = new Scanner(System.in);
System.out.print("Enter a number: ");
int limit = scanner.nextInt();
System.out.println("Prime numbers between 1 and " + limit + " are:");
for (int num = 2; num <= limit; num++) {
if (isPrime(num)) {
System.out.print(num + " ");
}
}
scanner.close();
}
public static boolean isPrime(int number) {
if (number <= 1) {
return false;
}
for (int i = 2; i <= Math.sqrt(number); i++) {
if (number % i == 0) {
return false;
}
}
return true;
}
}
```

Result/Output:

```
Output
Enter a number: 10
Prime numbers between 1 and 10 are:
2 3 5 7
=== Code Execution Successful ===
```

Experiment No: 4

Write a program to design a class account using the inheritance and static members which show all functions of a bank (Withdrawl, deposit)

```
class Account {
private static int accountCount = 0; // Static member to keep track of the number of
accounts
protected double balance; // Protected member to allow access in subclasses
public Account() {
accountCount++;
balance = 0.0;
}
public void deposit(double amount) {
if (amount > 0) {
balance += amount;
System.out.println("Deposited: Rs" + amount);
} else {
System.out.println("Deposit amount must be positive.");
} }
public void withdraw(double amount) {
if (amount > 0 && amount <= balance) {
balance -= amount;
System.out.println("Withdrew: Rs" + amount);
} else {
System.out.println("Insufficient balance or invalid amount.");
}
}
public double getBalance() {
return balance;
}
public static int getAccountCount() {
return accountCount;
}}
class SavingsAccount extends Account {
private double interestRate;
public SavingsAccount(double interestRate) {
super(); // Call the constructor of the superclass
this.interestRate = interestRate;
}
```

```
public void applyInterest() {
double interest = balance * interestRate / 100;
deposit(interest);
System.out.println("Interest applied: $" + interest);
}}
public class Bank {
public static void main(String[] args) {
SavingsAccount myAccount = new SavingsAccount(5.0); // 5% interest rate
myAccount.deposit(2000);
myAccount.withdraw(200);
myAccount.applyInterest();
System.out.println("Current Balance: Rs" + myAccount.getBalance());
System.out.println("Total Accounts Created: " + Account.getAccountCount());
}
}
```

Result/Output:

```
Output

Deposited: Rs 2000.0
Withdrew: Rs 200.0
Deposited: Rs 90.0
Interest applied: Rs 90.0
Current Balance: Rs 1890.00
Total Accounts Created: 1
```

Experiment No: 5

Write a program to create a simple class to find out the area and perimeter of rectangle using super and this keyword.

```
class Shape {
protected double length;
protected double width;
public Shape(double length, double width) {
this.length = length;
this.width = width;
}}
class Rectangle extends Shape {
public Rectangle(double length, double width) {
super(length, width); // Call the constructor of the superclass
}
public double area() {
return length * width; // Calculate area
}
public double perimeter() {
return 2 * (length + width); // Calculate perimeter
}}
public class RectangleDemo {
public static void main(String[] args) {
Rectangle rectangle = new Rectangle(5.0, 3.0); // Create a rectangle with length 5.0 and
width 3.0
System.out.println("Area of Rectangle: " + rectangle.area());
System.out.println("Perimeter of Rectangle: " + rectangle.perimeter());
}}
```

Result/Output:

```
Area of Rectangle: 15.0
Perimeter of Rectangle: 16.0
```

Experiment No: 6

Write a program to find the factorial of a given number using recursion.

```
import java.util.Scanner;
public class Factorial {
   // Recursive method to calculate factorial
   public static int factorial(int n) {
      // Base case: factorial of 0 or 1 is 1
      if (n == 0 || n == 1) {
         return 1;
      } else {
         return n * factorial(n - 1); // Recursive step
      }
   }

   public static void main(String[] args) {
      Scanner scanner = new Scanner(System.in);

      // Asking the user to input a number
      System.out.print("Enter a number: ");
      int number = scanner.nextInt();

      // Calling the recursive method and printing the result
      System.out.println("Factorial of " + number + " is: " + factorial(number));

      scanner.close();
   }
}
```

Result/Output:

```
Output

Enter a number: 5
Factorial of 5 is: 120

=== Code Execution Successful ===
```

Experiment No: 7

Write a program to design a class using abstract methods and abstract classes.

```
// Abstract class Shape with an abstract method area()
abstract class Shape {
   // Abstract method (does not have a body)
   public abstract double area();

   // Regular method
   public void display() {
      System.out.println("This is a shape.");
   }
}

// Subclass Circle that extends Shape
class Circle extends Shape {
   private double radius;

   // Constructor for Circle
   public Circle(double radius) {
      this.radius = radius;
   }

   // Implementing the abstract method area() for Circle
   public double area() {
      return Math.PI * radius * radius;
   }
}

// Subclass Rectangle that extends Shape
class Rectangle extends Shape {
   private double length, width;

   // Constructor for Rectangle
   public Rectangle(double length, double width) {
      this.length = length;
      this.width = width;
   }

   // Implementing the abstract method area() for Rectangle
   public double area() {
      return length * width;
   }
}

public class Main {
```

```
    public static void main(String[] args) {
        // Creating instances of Circle and Rectangle
        Shape circle = new Circle(6.0);   // Circle with radius 5
        Shape rectangle = new Rectangle(2.0, 6.0);   // Rectangle with length 4 and width 6

        // Display area for both shapes
        System.out.println("Area of Circle: " + circle.area());
        System.out.println("Area of Rectangle: " + rectangle.area());

        // Calling the regular method from Shape class
        circle.display();
        rectangle.display();
    }
}
```

Result/Output:

```
Output

Area of Circle: 113.09733552923255
Area of Rectangle: 12.0
This is a shape.
This is a shape.
```

Experiment No: 8

Write a program to count the number of objects created for a class using static member function

```
public class ObjectCounter {
   // Static variable to count the number of objects
   private static int count = 0;

   // Constructor that increments the count every time an object is created
   public ObjectCounter() {
      count++;
   }

   // Static method to get the current object count
   public static int getObjectCount() {
      return count;
   }

   public static void main(String[] args) {
      // Creating objects
      ObjectCounter obj1 = new ObjectCounter();
      ObjectCounter obj2 = new ObjectCounter();
      ObjectCounter obj3 = new ObjectCounter();

      // Getting the number of objects created
      System.out.println("Number of objects created: " + ObjectCounter.getObjectCount());
   }
}
```

Result/Output:

```
Output

Number of objects created: 3
```

Experiment No: 9

Write a program to demonstrate the use of function overloading.

```
class MathOperations {
// Method to add two integers
public int add(int a, int b) {
return a + b;
}
// Method to add three integers
public int add(int a, int b, int c) {
return a + b + c;
}
// Method to add two double values
public double add(double a, double b) {
return a + b;
}}
public class FunctionOverloadingDemo {
public static void main(String[] args) {
MathOperations mathOps = new MathOperations();
// Calling the overloaded add methods
int sum1 = mathOps.add(5, 10); // Calls the first method
int sum2 = mathOps.add(5, 10, 15); // Calls the second method
double sum3 = mathOps.add(5.5, 10.5); // Calls the third method
// Displaying the results
System.out.println("Sum of two integers: " + sum1);
System.out.println("Sum of three integers: " + sum2);
System.out.println("Sum of two doubles: " + sum3);
}}
```

Result/Output:

```
Sum of two integers: 15
Sum of three integers: 30
Sum of two doubles: 16.0
```

Experiment No: 10

Write a program to demonstrate the use of function overloading.

```
public class OverloadExample {

    // Method with one int parameter
    public void display(int a) {
        System.out.println("Displaying int: " + a);
    }

    // Method with one double parameter
    public void display(double a) {
        System.out.println("Displaying double: " + a);
    }

    // Method with two parameters
    public void display(String a, int b) {
        System.out.println("Displaying String and int: " + a + ", " + b);
    }

    public static void main(String[] args) {
        OverloadExample obj = new OverloadExample();

        obj.display(10);            // Calls display(int)
        obj.display(3.14);          // Calls display(double)
        obj.display("Age", 25);     // Calls display(String, int)
    }
}
```

Output

```
Displaying int: 10
Displaying double: 3.14
Displaying String and int: Age, 25
```

Experiment No: 11

Write a program to demonstrate the use of multiple inheritance

```
// First interface
interface Printable {
   void print();
}

// Second interface
interface Showable {
   void show();
}

// Class implementing both interfaces
class Demo implements Printable, Showable {
   public void print() {
      System.out.println("Printing from Printable interface");
   }

   public void show() {
      System.out.println("Showing from Showable interface");
   }
}

public class Main {
   public static void main(String[] args) {
      Demo obj = new Demo();
      obj.print();  // Calls print() from Printable
      obj.show();   // Calls show() from Showable
   }
}
```

Output

```
Printing from Printable interface
Showing from Showable interface

=== Code Execution Successful ===
```

Experiment No: 12

Write a program to design a string class that perform string method(Equal, Reverse the string, change case).

```
import java.util.Scanner;

class StringHandler {
   private String text;

   // Constructor
   public StringHandler(String text) {
      this.text = text;
   }

   // Method to check if another string is equal
   public boolean isEqual(String other) {
      return text.equals(other);
   }

   // Method to reverse the string
   public String reverse() {
      StringBuilder sb = new StringBuilder(text);
      return sb.reverse().toString();
   }

   // Method to change case of the string
   public String changeCase() {
      StringBuilder result = new StringBuilder();
      for (char c : text.toCharArray()) {
         if (Character.isUpperCase(c)) {
            result.append(Character.toLowerCase(c));
         } else if (Character.isLowerCase(c)) {
            result.append(Character.toUpperCase(c));
         } else {
            result.append(c); // Keep non-alphabet characters as is
         }
      }
      return result.toString();
   }
}

public class Main {
   public static void main(String[] args) {
      Scanner scanner = new Scanner(System.in);
```

```
        // Input original string
        System.out.print("Enter a string: ");
        String input = scanner.nextLine();

        // Create object
        StringHandler handler = new StringHandler(input);

        // Input second string for comparison
        System.out.print("Enter another string to compare: ");
        String compareTo = scanner.nextLine();

        // Perform operations
        System.out.println("Strings are equal: " + handler.isEqual(compareTo));
        System.out.println("Reversed string: " + handler.reverse());
        System.out.println("Case changed string: " + handler.changeCase());

        scanner.close();
    }
}
```

Output

```
Enter a string: dhruv
Enter another string to compare: dhruva
Strings are equal: false
Reversed string: vurhd
Case changed string: DHRUV

=== Code Execution Successful ===
```

Write a program to handle the exception using try and multiple catch block.

```
import java.util.Scanner;

public class MultipleCatchExample {
   public static void main(String[] args) {
      Scanner scanner = new Scanner(System.in);

      try {
         // Division operation
         System.out.print("Enter numerator: ");
         int num = scanner.nextInt();

         System.out.print("Enter denominator: ");
         int den = scanner.nextInt();

         int result = num / den;  // May throw ArithmeticException
         System.out.println("Result of division: " + result);

         // Array access operation
         int[] arr = {10, 20, 30};
         System.out.print("Enter array index to access (0-2): ");
         int index = scanner.nextInt();

         System.out.println("Value at index " + index + ": " + arr[index]); // May
throw ArrayIndexOutOfBoundsException

      } catch (ArithmeticException e) {
         System.out.println("Error: Cannot divide by zero.");
      } catch (ArrayIndexOutOfBoundsException e) {
         System.out.println("Error: Array index is out of range.");
      } catch (Exception e) {
         System.out.println("General error occurred: " + e.getMessage());
      } finally {
         System.out.println("Program execution finished.");
         scanner.close();
      }
   }
}
```

Output

```
Enter numerator: 23
Enter denominator: 0
ERROR!
Error: Cannot divide by zero.
Program execution finished.
```

Output

```
Enter numerator: 23
Enter denominator: 23
Result of division: 1
Enter array index to access (0-2): 3
ERROR!
Error: Array index is out of range.
Program execution finished.
```

Experiment No: 14

Write a program to create a package that access the member of External class as well as same package.

```
package GFG1;
// Creating Interface
interface GFG1Interface {
	String name = "This is the Interface of GF1";
	void GFG1Interface();
}
public class GFG1 {

	// Instance variable
	String name;

	// Getter Function
	public String getName() { return name; }

	// Setter Function
	public void setName(String name) { this.name = name; }
}
package GFG2;
// Creating Interface
interface GFG3Interface {
	String name = "GFG";
	public void interfaceGFG();
}
// Creating Abstract class
abstract class GFGabstract {
	String name = "GFGAbstract";

	// Abstract Method
	abstract public void print();
}
public class GFG3 {

	// Instance Variables
	int first;
	int second;

	// Creating Constructor
	GFG3(int a, int b)
	{
		this.first = a;
		this.second = b;
	}
```

```
        // Creating add Function
        public int add() { return this.first + this.second; }
}
package GFG2;
// Importing the members of GFG1 package
import GFG1.*;
public class GFG2 implements GFG3Interface {

        @Override public void interfaceGFG()
        {
                System.out.println(
                        "This is the interface of the GFG3class");
        }
        public static void main(String args[])
        {
                // Creating object of class GFG1
                GFG1 ob = new GFG1();

                // Calling setName Function
                ob.setName("GFGsetter");
                System.out.println(ob.getName());

                // Creating object of class GFG2
                GFG2 ob2 = new GFG2();

                ob2.interfaceGFG();
        }
}
```

Output:

```
GFGsetter
This is the interface of the GFG3class
```

Experiment No: 16

Write a program to handle the user defined exception using throw keyword.

```
// Define the user-defined exception class
class InvalidAgeException extends Exception {
   public InvalidAgeException(String message) {
      super(message);
   }
}

public class UserDefinedExceptionExample {

   // Method to check age and throw the user-defined exception
   public static void checkAge(int age) throws InvalidAgeException {
      if (age < 18) {
         throw new InvalidAgeException("Age must be 18 or older.");
      } else {
         System.out.println("Age is valid.");
      }
   }

   // Main method to test the exception
   public static void main(String[] args) {
      try {
         // Test with an invalid age
         checkAge(15);
      } catch (InvalidAgeException e) {
         System.out.println("Exception caught: " + e.getMessage());
      }

      try {
         // Test with a valid age
         checkAge(20);
      } catch (InvalidAgeException e) {
         System.out.println("Exception caught: " + e.getMessage());
      }
   }
}
```

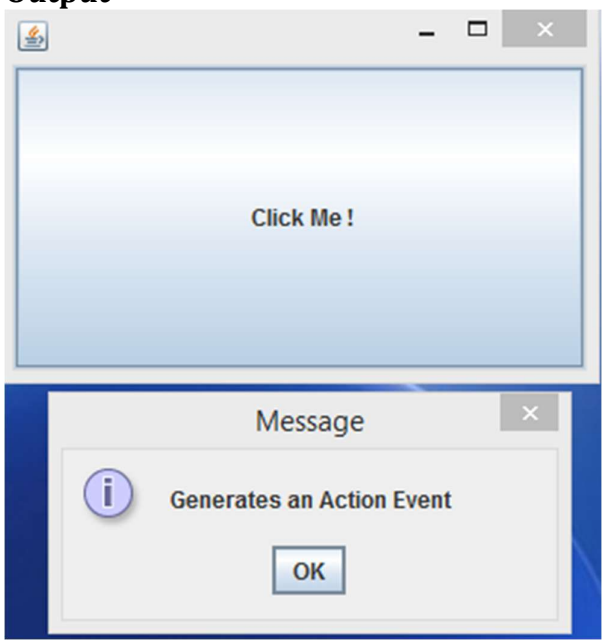
```
Exception caught: Age must be 18 or older.
Age is valid.
```

Experiment No: 17

Write a program to create a class component that shows controls and event handling on that controls.

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
public class EventListenerTest extends JFrame implements ActionListener {
  JButton button;
  public static void main(String args[]) {
    EventListenerTest object = new EventListenerTest();
    object.createGUI();
  }
  void createGUI() {
    button = new JButton(" Click Me !");
    setSize(300,200);
    setLocationRelativeTo(null);
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    setVisible(true);
    add(button);
    button.addActionListener(this);
  }
  public void actionPerformed(ActionEvent ae) {
    if(ae.getSource() == button) {
      JOptionPane.showMessageDialog(null, "Generates an Action Event");
    }
  }
}
```

Output



Experiment No: 18

```
import java.util.Scanner;

public class MathCalculator {

    public static void main(String[] args) {
        // Create a Scanner object to take input
        Scanner scanner = new Scanner(System.in);

        // Display the calculator options
        System.out.println("Simple Math Calculator");
        System.out.println("Choose an operation:");
        System.out.println("1. Addition");
        System.out.println("2. Subtraction");
        System.out.println("3. Multiplication");
        System.out.println("4. Division");

        // Get the user's choice of operation
        int choice = scanner.nextInt();

        // Ask for the two numbers to operate on
        System.out.println("Enter first number: ");
        double num1 = scanner.nextDouble();
        System.out.println("Enter second number: ");
        double num2 = scanner.nextDouble();

        double result = 0;

        // Perform the operation based on the user's choice
        switch (choice) {
            case 1: // Addition
                result = num1 + num2;
                System.out.println("Result: " + result);
                break;
            case 2: // Subtraction
                result = num1 - num2;
                System.out.println("Result: " + result);
                break;
            case 3: // Multiplication
                result = num1 * num2;
                System.out.println("Result: " + result);
                break;
            case 4: // Division
                if (num2 != 0) {
                    result = num1 / num2;
                    System.out.println("Result: " + result);
                } else {
                    System.out.println("Error: Division by zero is not allowed.");
```

```
            }
            break;
        default:
            System.out.println("Invalid choice, please select a valid operation.");
            break;
    }

    // Close the scanner
    scanner.close();
  }
}
```

```
Simple Math Calculator
Choose an operation:
1. Addition
2. Subtraction
3. Multiplication
4. Division
1
Enter first number:
2
Enter second number:
3
Result: 5.0
```

Experiment No: 19

Write a program to draw the line, Rectangle, oval, text using the graphics method.

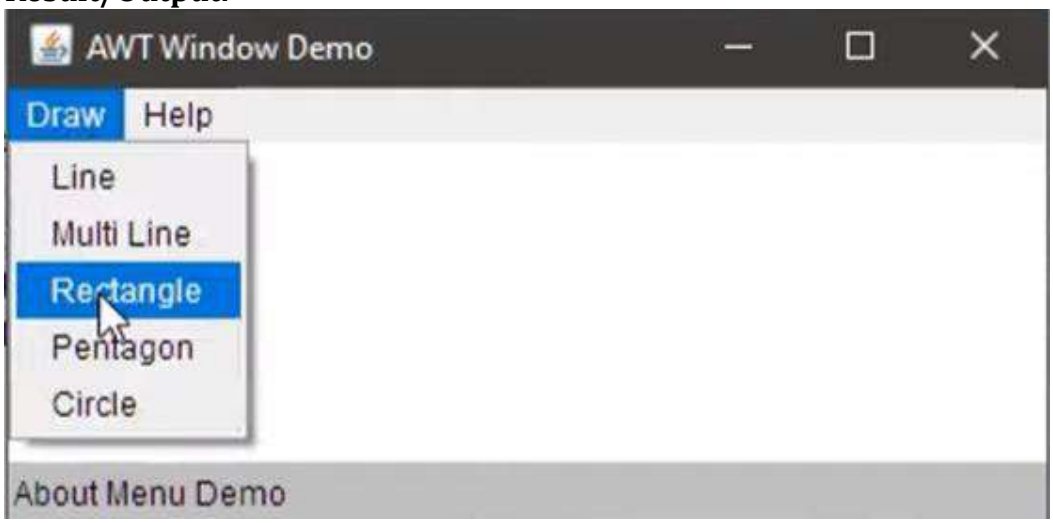
```
import javax.swing.*;
import java.awt.*;

public class GraphicsDemo extends JFrame {

   public GraphicsDemo() {
      setTitle("Graphics Drawing Example");
      setSize(400, 300);
      setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
      setVisible(true);
   }

   public void paint(Graphics g) {
      super.paint(g); // Call the superclass method to ensure proper painting

      // Draw a line from (50, 50) to (200, 50)
      g.drawLine(50, 50, 200, 50);

      // Draw a rectangle at (50, 80) with width 150 and height 60
      g.drawRect(50, 80, 150, 60);

      // Draw an oval inscribed inside the above rectangle
      g.drawOval(50, 80, 150, 60);

      // Draw a string at (50, 170)
      g.drawString("Hello, Graphics!", 50, 170);
   }

   public static void main(String[] args) {
      new GraphicsDemo();
   }
}
```

Experiment No: 20

Write a program to create a menu using the frame.

```
MenuItem("Rectangle");
MenuItem pentagon = new MenuItem("Pentagon");
MenuItem circle = new MenuItem("Circle");
// Add items to "Draw"
drawMenu.add(line);
drawMenu.add(multiLine);
drawMenu.add(rectangle);
drawMenu.add(pentagon);
drawMenu.add(circle);
// Add action listeners
line.addActionListener(this);
multiLine.addActionListener(this);
rectangle.addActionListener(this);
pentagon.addActionListener(this);
circle.addActionListener(this);
// Create "Help" menu
Menu helpMenu = new Menu("Help");
// Add menus to the menu bar
menuBar.add(drawMenu);
menuBar.add(helpMenu);
// Set menu bar to the frame
setMenuBar(menuBar);
// Frame settings
setTitle("AWT Window Demo");
setSize(400, 300);
setLayout(null);
setVisible(true);
// Add window close handler
addWindowListener(new WindowAdapter() {
public void windowClosing(WindowEvent e) {
dispose();
}
});
} public void actionPerformed(ActionEvent e) {
System.out.println(e.getActionCommand() + " selected");
}
public static void main(String[] args) {
new AWTMenuExample();
}
}
```

Result/Output:



Experiment No: 21

Write a program to implement the flow layout and border layout.

```
// Java program to illustrate the BorderLayout
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;

// class extends JFrame
class BoderLayoutDemo extends JFrame {

	BoderLayoutDemo()
	{

		// Creating Object of Jpanel class
		JPanel pa = new JPanel();

		// set the layout
		pa.setLayout(new BorderLayout());

		// add a new JButton with name "wel" and it is
		// lie top of the container
		pa.add(new JButton("WelCome"), BorderLayout.NORTH);

		// add a new JButton with name "come" and it is
		// lie button of the container
		pa.add(new JButton("Geeks"), BorderLayout.SOUTH);

		// add a new JButton with name "Layout" and it is
		// lie left of the container
		pa.add(new JButton("Layout"), BorderLayout.EAST);

		// add a new JButton with name "Border" and it is
		// lie right of the container
		pa.add(new JButton("Border"), BorderLayout.WEST);

		// add a new JButton with name "hello everybody" and it is
		// lie center of the container
		pa.add(new JButton("GeeksforGeeks"), BorderLayout.CENTER);

		// add the pa object which refer to the Jpanel
		add(pa);

		// Function to close the operation of JFrame.
		setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

		// Function to set size of JFrame.
		setSize(300, 300);
```

```
            // Function to set visible status of JFrame.
            setVisible(true);
        }
}

class MainFrame {

        // Driver code
        public static void main(String[] args)
        {

            // calling the constructor
            new BoderLayoutDemo();
        }
}
```

Experiment No: 22

Write a program to implement the gridLayout, cardLayout

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

public class LayoutExample {

   public static void main(String[] args) {
      // Create the frame
      JFrame frame = new JFrame("GridLayout and CardLayout Example");
      frame.setSize(400, 300);
      frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

      // Create a panel for GridLayout
      JPanel gridPanel = new JPanel(new GridLayout(2, 2));  // 2x2 grid layout

      // Add buttons to the GridLayout panel
      gridPanel.add(new JButton("Button 1"));
      gridPanel.add(new JButton("Button 2"));
      gridPanel.add(new JButton("Button 3"));
      gridPanel.add(new JButton("Button 4"));

      // Create a panel for CardLayout
      JPanel cardPanel = new JPanel();
      CardLayout cardLayout = new CardLayout();
      cardPanel.setLayout(cardLayout);

      // Create panels for each card
      JPanel card1 = new JPanel();
      card1.add(new JLabel("Card 1"));
      card1.add(new JButton("Button on Card 1"));

      JPanel card2 = new JPanel();
      card2.add(new JLabel("Card 2"));
      card2.add(new JButton("Button on Card 2"));

      JPanel card3 = new JPanel();
      card3.add(new JLabel("Card 3"));
      card3.add(new JButton("Button on Card 3"));

      // Add cards to the CardLayout panel
      cardPanel.add(card1, "Card 1");
```

```
        cardPanel.add(card2, "Card 2");
        cardPanel.add(card3, "Card 3");

        // Create a panel for navigation between cards
        JPanel cardNavigationPanel = new JPanel();
        JButton showCard1 = new JButton("Show Card 1");
        JButton showCard2 = new JButton("Show Card 2");
        JButton showCard3 = new JButton("Show Card 3");

        // Add action listeners to the buttons for switching cards
        showCard1.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                cardLayout.show(cardPanel, "Card 1");
            }
        });
        showCard2.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                cardLayout.show(cardPanel, "Card 2");
            }
        });
        showCard3.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                cardLayout.show(cardPanel, "Card 3");
            }
        });

        // Add navigation buttons to the navigation panel
        cardNavigationPanel.add(showCard1);
        cardNavigationPanel.add(showCard2);
        cardNavigationPanel.add(showCard3);

        // Create a split pane to display both GridLayout and CardLayout panels side
by side
        JSplitPane splitPane = new JSplitPane(JSplitPane.HORIZONTAL_SPLIT,
gridPanel, cardPanel);
        frame.getContentPane().add(splitPane, BorderLayout.CENTER);

        // Add the navigation panel below the card panel
        frame.getContentPane().add(cardNavigationPanel, BorderLayout.SOUTH);

        // Set frame visibility
        frame.setVisible(true);
    }
}
```

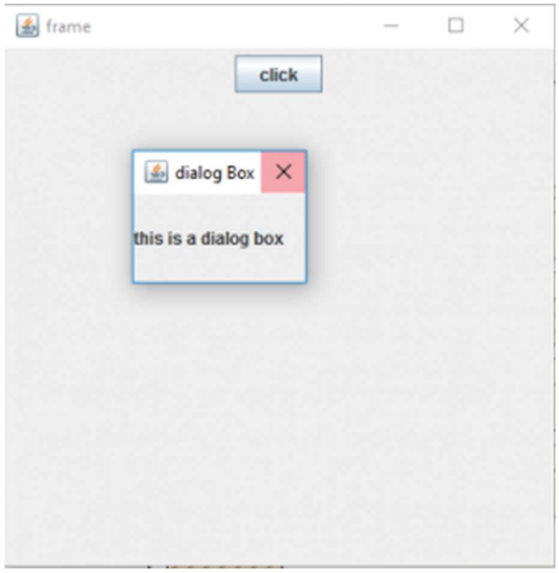
Experiment No: 23

Write a program to create a dialogbox.

```
// java Program to create a simple JDialog
import java.awt.event.*;
import java.awt.*;
import javax.swing.*;
class solve extends JFrame implements ActionListener {

        // frame
        static JFrame f;

        // main class
        public static void main(String[] args)
        {
                // create a new frame
                f = new JFrame("frame");

                // create a object
                solve s = new solve();

                // create a panel
                JPanel p = new JPanel();

                JButton b = new JButton("click");

                // add actionlistener to button
                b.addActionListener(s);

                // add button to panel
                p.add(b);
```

```
            f.add(p);

            // set the size of frame
            f.setSize(400, 400);

            f.show();
        }
        public void actionPerformed(ActionEvent e)
        {
            String s = e.getActionCommand();
            if (s.equals("click")) {
                // create a dialog Box
                JDialog d = new JDialog(f, "dialog Box");

                // create a label
                JLabel l = new JLabel("this is a dialog box");

                d.add(l);

                // setsize of dialog
                d.setSize(100, 100);

                // set visibility of dialog
                d.setVisible(true);
            }
        }
}
```

Output:



Experiment No: 24

Write a program to create Frame that display the student information

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
import java.io.*;

public class GFG {

	// Function to write a student
	// information in JFrame and
	// storing it in a file
	public static void StudentInfo()
	{

		// Creating a new frame using JFrame
		JFrame f
			= new JFrame(
				"Student Details Form");

		// Creating the labels
		JLabel l1, l2, l3, l4, l5;

		// Creating three text fields.
		// One for student name, one for
		// college mail ID and one
		// for Mobile No
		JTextField t1, t2, t3;

		// Creating two JComboboxes
		// one for Branch and one
		// for Section
		JComboBox j1, j2;

		// Creating two buttons
		JButton b1, b2;

		// Naming the labels and setting
		// the bounds for the labels
		l1 = new JLabel("Student Name:");
		l1.setBounds(50, 50, 100, 30);
		l2 = new JLabel("College Email ID:");
		l2.setBounds(50, 120, 120, 30);
		l3 = new JLabel("Branch:");
		l3.setBounds(50, 190, 50, 30);
		l4 = new JLabel("Section:");
		l4.setBounds(420, 50, 70, 30);
```

```
l5 = new JLabel("Mobile No:");
l5.setBounds(420, 120, 70, 30);

// Creating the textfields and
// setting the bounds for textfields
t1 = new JTextField();
t1.setBounds(150, 50, 130, 30);
t2 = new JTextField();
t2.setBounds(160, 120, 130, 30);
t3 = new JTextField();
t3.setBounds(490, 120, 130, 30);

// Creating two string arrays one for
// braches and other for sections
String s1[]
        = { " ", "CSE", "ECE", "EEE",
                "CIVIL", "MEC", "Others" };
String s2[]
        = { " ", "Section-A", "Section-B",
                "Section-C", "Section-D",
                "Section-E" };

// Creating two JComboBoxes one for
// selecting branch and other for
// selecting the section
// and setting the bounds
j1 = new JComboBox(s1);
j1.setBounds(120, 190, 100, 30);
j2 = new JComboBox(s2);
j2.setBounds(470, 50, 140, 30);

// Creating one button for Saving
// and other button to close
// and setting the bounds
b1 = new JButton("Save");
b1.setBounds(150, 300, 70, 30);
b2 = new JButton("close");
b2.setBounds(420, 300, 70, 30);

// Adding action listener
b1.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent e)
        {

                // Getting the text from text fields
                // and JComboboxes
                // and copying it to a strings
```

```
                String s1 = t1.getText();
                String s2 = t2.getText();
                String s3 = j1.getSelectedItem() + "";
                String s4 = j2.getSelectedItem() + "";
                String s5 = t3.getText();
                if (e.getSource() == b1) {
                        try {

                                // Creating a file and
                                // writing the data
                                // into a Textfile.
                                FileWriter w
                                        = new FileWriter(
                                                "GFG.txt", true);

                                w.write(s1 + "\n");
                                w.write(s2 + "\n");
                                w.write(s3 + "\n");
                                w.write(s4 + "\n");
                                w.write(s5 + "\n");
                                w.close();
                        }
                        catch (Exception ae) {
                                System.out.println(ae);
                        }
                }

                // Shows a Pop up Message when
                // save button is clicked
                JOptionPane
                        .showMessageDialog(
                                f,
                                "Successfully Saved"
                                        + " The Details");
        }
});

// Action listener to close the form
b2.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent e)
        {
                f.dispose();
        }
});

// Default method for closing the frame
f.addWindowListener(new WindowAdapter() {
        public void windowClosing(WindowEvent e)
```

```
                {
                        System.exit(0);
                }
        });

        // Adding the created objects
        // to the frame
        f.add(l1);
        f.add(t1);
        f.add(l2);
        f.add(t2);
        f.add(l3);
        f.add(j1);
        f.add(l4);
        f.add(j2);
        f.add(l5);
        f.add(t3);
        f.add(b1);
        f.add(b2);
        f.setLayout(null);
        f.setSize(700, 600);
        f.setVisible(true);
    }
    // Driver code
    public static void main(String args[])
    {
        StudentInfo();
    }
}
```