```
// prac-1a  static method , constructor , method overloading
//
--------------------------------------------------------------------------------
// #code
// -----
class Example{
    int a,b;
    public Example()
    {
    this.a =0;
    this.b =0;
    System.out.println("Default Constructor : a ="+a+",b="+b);
    }
    public Example(int a)
    {
    this.a=a;
    this.b=0;
    System.out.println("Single parameter Constructor : a ="+a+",b="+b);
    }
    public Example(int a, int b)
    {
    this.a=a;
    this.b=b;
    System.out.println("two parameter Constructor : a ="+a+",b="+b);
    }
    public void display(){
    System.out.println("Display with no parameters: a ="+a+",b="+b);
    }
    public void display(int a){
    System.out.println("Display with one parameters: a ="+a+",b="+b);
    }
    public void display(int a, int b){
    System.out.println("Display with two parameters: a ="+a+",b="+b);
    }
    public static void staticMethod(){
    System.out.println("this is a static method");
    }
    public static void main(String[] args)
    {
    Example obj1 = new Example();
    Example obj2 = new Example(5);
    Example obj3 = new Example(5,10);
    obj1.display();
    obj1.display(7);
    obj1.display(7,14);
    Example.staticMethod();
    }
    }

    // o/p

    // PS C:\Users\Admin\Desktop\sycs-41\prac3> javac Example.java
    // PS C:\Users\Admin\Desktop\sycs-41\prac3> java Example
    // Default Constructor : a =0,b=0
    // Single parameter Constructor : a =5,b=0
    // two parameter Constructor : a =5,b=10
    // Display with no parameters: a =0,b=0
    // Display with one parameters: a =7,b=0
    // Display with two parameters: a =7,b=14
    // this is a static method


    // ##prac-1b
// ------------
// wap to implement the concept of inheritance and method overriding
// -------------------------------------------------------------------
// #code
// -----
class A
{
 void show()
{
System.out.println("base class");
}
}
class B extends A
{
void show()
{
System.out.println("Derived Class");
}
}
 class pr1b
{
public static void main(String[] args)
{
B s= new B();
A s1= new A();
s.show();
s1.show();
}
}

// o/p
// ----
// PS C:\Users\Admin\Desktop\sycs-41\prac3> javac pr1b.java
// PS C:\Users\Admin\Desktop\sycs-41\prac3> java pr1b
// Derived Class
// base class


// ##prac-2a
// ------------
// wap implement the concept of abstract classes and methods
// -------------------------------------------------------------
// #code
// ------
abstract class shape
{
public abstract double area();
}
class circle extends shape
{
 private double radius;
public circle(double radius)
{
 this.radius=radius;
}
//@override
public double area()
{
return Math.PI*radius*radius;
}
}
class pr2a
{
public static void main(String[] args)
{
```

```
circle c=new circle(10.0);
System.out.println("circle area is : "+c.area());
}
}

// o/p
// ----
// PS C:\Users\Admin\Desktop\sycs-41\prac3> javac pr2a.java
// PS C:\Users\Admin\Desktop\sycs-41\prac3> java pr2a
// circle area is : 314.1592653589793
//
--------------------------------------------------------------------------------


// ##2b
// -----------
// write a program to impliement interface
// _________________________________________
// #code
interface shape
{
 double area();
 double perimeter();
}
class circle implements shape
{
private double ra;
public circle(double ra)
{
this.ra=ra;
}
public double area()
{
return Math.PI*ra*ra;
}
public double perimeter()
{
return 2*Math.PI*ra*ra;
}
}
public class pr2b
{
 public static void main(String args[])
 {
 circle c=new circle(10.0);
 System.out.println("Area of circle is "+c.area());
 System.out.println("circle perimeter is "+c.perimeter());
 }
}

// o/p
// PS C:\Users\Admin\Desktop\sycs-41\prac4> javac pr2b.java
// PS C:\Users\Admin\Desktop\sycs-41\prac4> java pr2b
// Area of circle is 314.1592653589793
// circle perimeter is 628.3185307179587
//
________________________________________________________________________________


// ##3A
// ----
// write a program to define userdefine Exception and raise them as per requirement
//
```

```
// _
// #code

import java.util.*;
class CustomException extends Exception
{
 public CustomException(String message)
{
super(message);
}
}
public class pr3a
{
 public static void main(String args[])
{
 try
{
int age=20;
if(age<0)
{
 throw new CustomException("Age cannot be negative");
}
System.out.println("age "+age);
}
catch(CustomException e)
{
 System.out.println(e.getMessage());
}
}
}

// o/p
// PS C:\Users\Admin\Desktop\sycs-41\prac4> javac pr3a.java
// PS C:\Users\Admin\Desktop\sycs-41\prac4> java pr3a
// Age cannot be negative
// ____________________
// //if age positive if age=20;
// --------------------
// PS C:\Users\Admin\Desktop\sycs-41\prac4> javac pr3a.java
// PS C:\Users\Admin\Desktop\sycs-41\prac4> java pr3a
// age 20
//
```

```
// ###3B
// ------
// PREDEFINE EXCEPTION
// ______________________

public class pr3b
{
 public static void main(String[] args)
 {
 try
 {
 int result=divide(10,0);
 System.out.println("result is :"+result);
 }
 catch(ArithmeticException e)
 {
 System.out.println("Error : Division by Zero");
 }
 }
 public static int divide(int a, int b)
 {
 return a/b;
 }
}


// o/p
// PS C:\Users\Admin\Desktop\sycs-41\prac4> javac pr3b.java
// PS C:\Users\Admin\Desktop\sycs-41\prac4> java pr3b
// Error : Division by Zero
//
```

```
// ##prac4a
// --------
// write program to demonstrate the method of
// a.list interface
// b.set interface
// c.map interface
// ________________________________
// A.list interface
// #code
// _____


import java.util.*;
class ListDemo {
 public static void main(String[] args) {
 List<String> fruits = new ArrayList<>();
 fruits.add("Apple");
 fruits.add("Banana");
 fruits.add("Cherry");
 fruits.add("Kiwi");
 fruits.add("Banana");
 fruits.add("Mango");
 System.out.println("List of fruits");
 for (String fruit : fruits) {
 System.out.println(fruit);
 }
 System.out.println("\n Element at index 2 :" + fruits.get(2));
 fruits.remove("Banana");
 System.out.println("\n list after removing Banana :" + fruits);
 System.out.println("\n list Contained Mango :" +
fruits.contains("Mango"));
 System.out.println("\n iterating using listiterator");
 ListIterator<String> iterator = fruits.listIterator();
 while (iterator.hasNext()) {
 System.out.println(iterator.next());
 }
 }
}

// o/p
// ----
// PS C:\Users\Admin\Desktop\sycs-41\prac4> javac ListDemo.java
// PS C:\Users\Admin\Desktop\sycs-41\prac4> java ListDemo
// List of fruits
// Apple
// Banana
// Cherry
// Kiwi
// Banana
// Mango
//  Element at index 2 :Cherry
//  list after removing Banana :[Apple, Cherry, Kiwi, Banana, Mango]
//  list Contained Mango :true
//  iterating using listiterator
// Apple
// Cherry
// Kiwi
// Banana
// Mango
// ___________________________________________________________________


// #4B. SET INTERFACE
// __________________
// #code
// -------

import java.util.*;
public class SetDemo1
{
public static void main(String args[])
{
 Set<String> fruits = new HashSet<>();
 fruits.add("Apple");
 fruits.add("mango");
 fruits.add("banana");
 fruits.add("orange");
 fruits.add("Apple");
 System.out.println("Set of Fruits :");
 for (String fruit : fruits )
 {
 System.out.println(fruit);
 }
 System.out.println("\nSet Contains 'Mango' :"+fruits.contains("mango"));
fruits.remove("banana");
System.out.println("\nSet after removing 'banana' :" +fruits);
 System.out.println("\nIterating using Iterator :");
 Iterator<String> iterator = fruits.iterator();
 while (iterator.hasNext())
 {
 System.out.println(iterator.next());
}
fruits.clear();
System.out.println("\nSet after clearing :" +fruits);
}
}

// o/p
// PS C:\Users\Admin\Desktop\SYCS-41\java> javac SetDemo1.java
// PS C:\Users\Admin\Desktop\SYCS-41\java> java SetDemo1
// Set of Fruits :
// banana
// orange
// Apple
// mango
// Set Contains 'Mango' :true
// Set after removing 'banana' :[orange, Apple, mango]
// Iterating using Iterator :
// orange
// Apple
// mango
// Set after clearing :[]
//
```

```
// ##4C Map interface
// ____________________
// #code

import java.util.*;
public class MapDemo
{
public static void main(String args[])
{
 Map<Integer,String> map = new HashMap<>();
 map.put(1, "Apple");
 map.put(2, "Banana");
 map.put(3, "Orange");
 map.put(4, "Mango");
 map.put(5, "Grapes");
 map.put(1, "cherry");
System.out.println("Map of Fruits :");
for (Map.Entry<Integer, String> entry : map.entrySet())
{
System.out.println("Key :"+entry.getKey() + ", Value : " + entry.getValue());
}
System.out.println("\n value for key 2 : "+map.get(2));
map.remove(3);
System.out.println("\n after removing key 3 : ");
for (Map.Entry<Integer, String> entry : map.entrySet())
 {
 System.out.println("Key :"+entry.getKey() + ", Value : " +
entry.getValue());
 }
 System.out.println("\nMap contains key 4 : "+map.containsKey(4));
System.out.println("map contains value 'Banana' :"+map.containsValue("Banana"));
 System.out.println("\n Iterating using forEach method :");
map.forEach((key, value) ->
 System.out.println("Key :"+key + ", Value :"+value));
map.clear();
 System.out.println("\nMap after clearing :"+map);
}
}

// o/p
// PS C:\Users\Admin\Desktop\SYCS-41\java> javac MapDemo.java
// PS C:\Users\Admin\Desktop\SYCS-41\java> java MapDemo
// Map of Fruits :
// Key :1, Value : cherry
// Key :2, Value : Banana
// Key :3, Value : Orange
// Key :4, Value : Mango
// Key :5, Value : Grapes
// value for key 2 : Banana
// after removing key 3 :
// Key :1, Value : cherry
// Key :2, Value : Banana
// Key :4, Value : Mango
// Key :5, Value : Grapes
// Map contains key 4 : true
// map contains value 'Banana' : true
// Iterating using forEach method :
// Key :1, Value :cherry
// Key :2, Value :Banana
// Key :4, Value :Mango
// Key :5, Value :Grapes
// Map after clearing :{}
// Pactical No.5 QWRITE promgram using various swing component design java
// application to accept student resume
// ____________________
// code:::
// ________
import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
public class StudentResumeForm extends JFrame {

private JLabel nameLabel,addressLabel,phoneLabel,emailLabel,educationLabel;
private JTextField nameField,addressField,phoneField,emailField;
private JTextArea addressArea, educationArea;
private JButton submitButton, resetButton;
public StudentResumeForm(){

 setTitle("Student Resume From");
 setSize(500,600);
 setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
 setLayout(new GridLayout(10,2,5,5));

 nameLabel=new JLabel("Name :");
 addressLabel=new JLabel("Address :");
 phoneLabel=new JLabel("Phone Number :");
 emailLabel=new JLabel("Email :");
 educationLabel=new JLabel("Education");
 nameField = new JTextField();
 addressArea = new JTextArea(3, 20);
 phoneField = new JTextField();
 emailField = new JTextField();
 educationArea = new JTextArea(3, 20);
 submitButton = new JButton("Submit");
 resetButton = new JButton("Reset");

 add(nameLabel);
 add(nameField);
 add(addressLabel);
 add(new JScrollPane(addressArea));
 add(phoneLabel);
 add(phoneField);
 add(emailLabel);
 add(emailField);
 add(educationLabel);
 add(new JScrollPane(educationArea));
 add(submitButton);
 add(resetButton);


 submitButton.addActionListener(new ActionListener() {
 @Override
 public void actionPerformed(ActionEvent e) {

 JOptionPane.showMessageDialog(null,"Resume submit successfully!");
 }
});
 resetButton.addActionListener(new ActionListener() {
 @Override
 public void actionPerformed(ActionEvent e) {

 nameField.setText("");
 addressArea.setText("");
 phoneField.setText("");
 emailField.setText("");
 educationArea.setText("");
 } });

setVisible(true);

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


// Practical No.06
// _______________
// _________________________________________________________
// #Practical No.06A - write a jdbc program that display data of given table
// ____________________
// code...
// _________
```

```
mysql> show tables;
+------------------+
| Tables_in_sycs41 |
+------------------+
| student |
+------------------+
1 row in set (0.02 sec)
mysql> desc student;
+--------------+--------------+------+-----+---------+-------+
| Field | Type | Null | Key | Default | Extra |
+--------------+--------------+------+-----+---------+-------+
| roll | int(11) | NO | PRI | NULL | |
| name | varchar(100) | NO | | NULL | |
| address | varchar(255) | YES | | NULL | |
| phone_number | varchar(15) | YES | | NULL | |
+--------------+--------------+------+-----+---------+-------+
4 rows in set (0.02 sec)
mysql> INSERT INTO student (roll, name, address, phone_number) VALUES
 -> (1, 'John Doe', '123 Elm St, Springfield', '555-1234'),
 -> (2, 'Jane Smith', '456 Oak St, Springfield', '555-5678'),
 -> (3, 'Alice Johnson', '789 Pine St, Springfield', '555-8765'),
 -> (4, 'Bob Brown', '101 Maple St, Springfield', '555-4321');
```

```
 ##code

 import java.sql.*;

public class DisplayTableData {
    public static void main(String[] args) {
        try {
            Class.forName("com.mysql.cj.jdbc.Driver");
            Connection conn = DriverManager.getConnection(
                "jdbc:mysql://localhost:3306/sycs?
useSSL=false&serverTimezone=UTC",
                "admin",
                "12345"
            );
            Statement statement = conn.createStatement();
            ResultSet resultSet = statement.executeQuery("SELECT * FROM
student");
            int columnCount = resultSet.getMetaData().getColumnCount();
            while (resultSet.next()) {
                for (int i = 1; i <= columnCount; i++) {
                    System.out.print(resultSet.getString(i) + "\t");
                }
                System.out.println();
            }
            resultSet.close();
            statement.close();
            conn.close();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```