M.SAI ROHAN

CH.SC.U4CSE24040

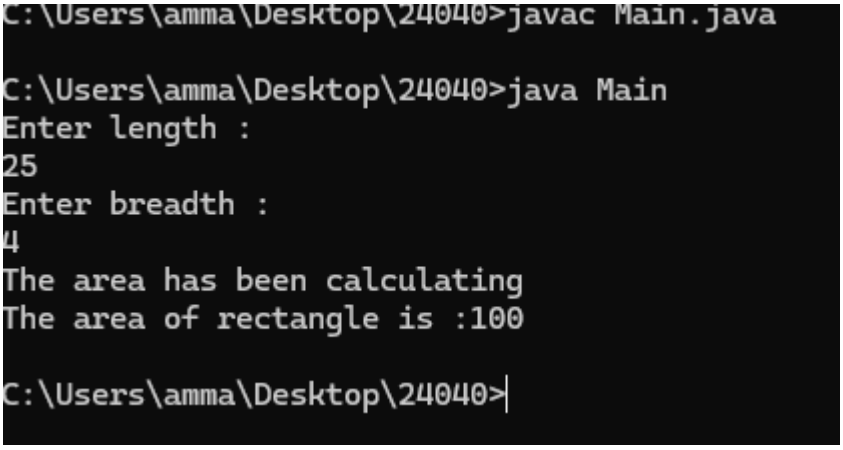
INHERITANCE

JAVA SINGLE INHERITANCE

1) Code:

```
import java.util.Scanner;
class Shape{
  public void calculateArea(){
    System.out.println("The area has been calculating");
  }
}
class Rectangle extends Shape{
  int length,breadth;
  Rectangle(int length,int breadth){
    this.length=length;
    this.breadth=breadth;
  }
  public void getArea(){
    System.out.println("The area of rectangle is :"+(length*breadth));
  }
}
public class Main{
  public static void main(String [] args){
    Scanner obj = new Scanner(System.in);
    System.out.println("Enter length :");
    int length=obj.nextInt();
    System.out.println("Enter breadth :");
    int breadth=obj.nextInt();
    Shape myShape=new Shape();
     myShape.calculateArea();
    Rectangle myObj=new Rectangle(length,breadth);
     myObj.getArea();

  }}
```

OUTPUT:

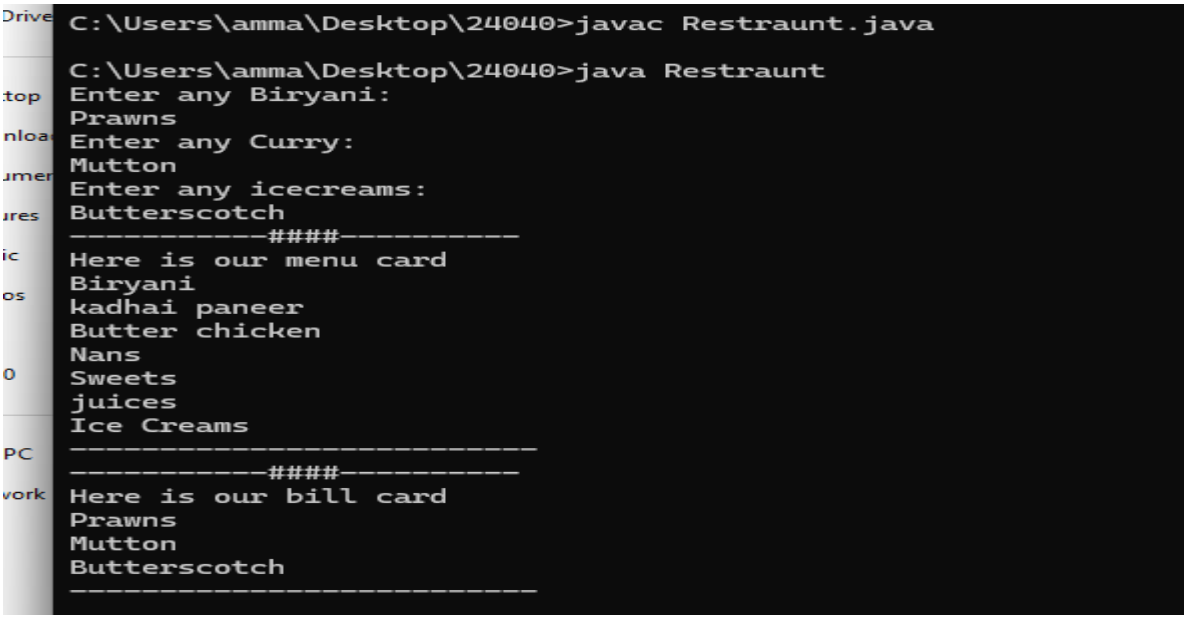


2)CODE:

```
import java.util.Scanner;
class RestrauntMenu{
   public void defaultMenu(){
       System.out.println("----------####----------");
      System.out.println("Here is our menu card");
      System.out.println("Biryani");
       System.out.println("kadhai paneer");
      System.out.println("Butter chicken");
       System.out.println("Nans");
      System.out.println("Sweets");
       System.out.println("juices");
      System.out.println("Ice Creams");
       System.out.println("-------------------------");
   }
}
class PaymentBill extends RestrauntMenu{
   String Biryani,curry,icecreams;
   PaymentBill(String Biryani,String curry,String icecreams){
      this.Biryani=Biryani;
      this.curry=curry;
      this.icecreams=icecreams;
   }
```

```
   public void yourBill(){
      System.out.println("-----------####----------");
        System.out.println("Here is our bill card");
       System.out.println(Biryani);
       System.out.println(curry);
      System.out.println(icecreams);
      System.out.println("--------------------------");
   }
}
public class Restraunt{
   public static void main(String [] args){
      Scanner obj=new Scanner(System.in);
      System.out.println("Enter any Biryani:");
      String Biryani=obj.nextLine();
      System.out.println("Enter any Curry:");
       String curry=obj.nextLine();
       System.out.println("Enter any icecreams:");
      String icecreams=obj.nextLine();
      RestrauntMenu myRes=new RestrauntMenu();
      myRes.defaultMenu();
      PaymentBill myobj=new PaymentBill(Biryani,curry,icecreams);
      Myobj.yourBill();
   }
}
```

OUTPUT:



JAVA HERIECHIAL INHERITANCE

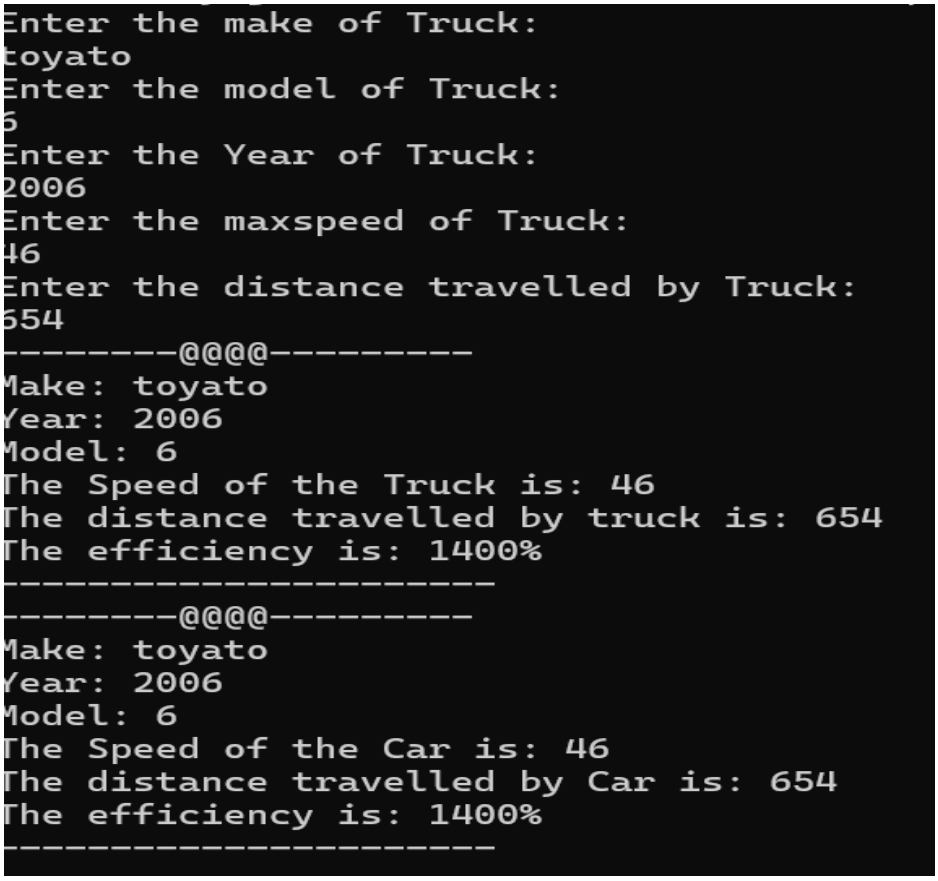
1)Code:

```
import java.util.Scanner;


class Vehicle {
   String make;
   int model, year, distance, maxspeed, efficiency;
   public Vehicle(String make, int model, int year, int distance, int maxspeed) {
      this.make = make;
      this.model = model;
      this.year = year;
      this.distance = distance;
      this.maxspeed = maxspeed;
   }
   public void special() {
      if (maxspeed != 0) {
         efficiency = (distance / maxspeed) * 100;
      } else {
```

```
            efficiency = 0;
        }
    }
}

class Truck extends Vehicle {
    public Truck(String make, int model, int year, int distance, int maxspeed) {
        super(make, model, year, distance, maxspeed);
        special();
    }
    public void displayTruckInfo() {
        System.out.println("--------@@@@---------");
        System.out.println("Make: " + make);
        System.out.println("Year: " + year);
        System.out.println("Model: " + model);
        System.out.println("The Speed of the Truck is: " + maxspeed);
        System.out.println("The distance travelled by truck is: " + distance);
        System.out.println("The efficiency is: " + efficiency + "%");
        System.out.println("----------------------");
    }
}

class Car extends Vehicle {
     Car(String make, int model, int year, int distance, int maxspeed) {
        super(make, model, year, distance, maxspeed);
        special();
    }
    public void displayCarInfo() {
        System.out.println("--------@@@@---------");
        System.out.println("Make: " + make);
        System.out.println("Year: " + year);
```

```
        System.out.println("Model: " + model);
        System.out.println("The Speed of the Car is: " + maxspeed);
        System.out.println("The distance travelled by Car is: " + distance);
        System.out.println("The efficiency is: " + efficiency + "%");
        System.out.println("----------------------");
    }
}

public class Details {
    public static void main(String[] args) {
        Scanner obj = new Scanner(System.in);
        System.out.println("Enter the make of Truck:");
        String make = obj.nextLine();
        System.out.println("Enter the model of Truck:");
        int model = obj.nextInt();
        System.out.println("Enter the Year of Truck:");
        int year = obj.nextInt();
        System.out.println("Enter the maxspeed of Truck:");
        int maxspeed = obj.nextInt();
        System.out.println("Enter the distance travelled by Truck:");
        int distance = obj.nextInt();
        Truck obj1 = new Truck(make, model, year, distance, maxspeed);
        obj1.displayTruckInfo();
        Car obj2 = new Car(make, model, year, distance, maxspeed);
        obj2.displayCarInfo();
    }
}
```

OUTPUT:



2) CODE:

class Shape { double area; double perimeter;

```
public Shape() {
    this.area = 0.0;
    this.perimeter = 0.0;
}


public void calculateArea() {
    System.out.println("The area is: " + area);
}


public void calculatePerimeter() {
    System.out.println("The perimeter is: " + perimeter);
}


}
```

class Circle extends Shape { int radius

```
Circle(int radius) {
    this.radius = radius;
}
```

```
public void calculateArea() {
    area = Math.PI * radius * radius;
    System.out.println("The area of Circle is: " + area);
}

public void calculatePerimeter() {
    perimeter = 2 * Math.PI * radius;
    System.out.println("The perimeter of Circle is: " + perimeter);
}


}
```

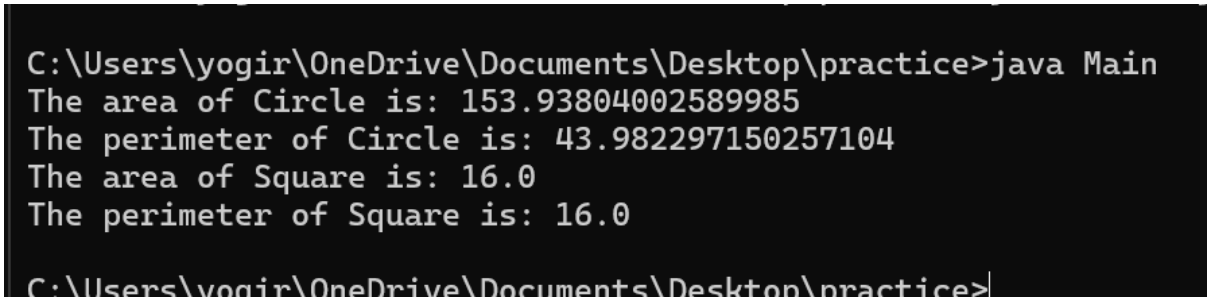
class Square extends Shape { int length;

```
Square(int length) {
    this.length = length;
}
public void calculateArea() {
    area = length * length;
    System.out.println("The area of Square is: " + area);
}
public void calculatePerimeter() {
    perimeter = 4 * length;
    System.out.println("The perimeter of Square is: " + perimeter);
}


}
```

public class Main { public static void main(String[] args) { e Circle circle = new Circle(7); circle.calculateArea(); circle.calculatePerimeter();

```
    Square square = new Square(4);
    square.calculateArea();
    square.calculatePerimeter();
}


}
```

OUTPUT:



MULTILEVEL INHERITANCE

1)Code:

```
class Pen {
  String color;
  String inkType;

  public Pen(String color, String inkType) {
    this.color = color;
    this.inkType = inkType;
  }

  public void write() {
    System.out.println("Writing with a " + color + " pen using " + inkType + " ink.");
  }
}

class FountainPen extends Pen {
  String nibSize;

  public FountainPen(String color, String inkType, String nibSize) {
    super(color, inkType);
    this.nibSize = nibSize;
  }

  public void write() {
    System.out.println("Writing with a " + color + " fountain pen using " + inkType + " ink with a " + nibSize + " nib.");
  }

  public void refill() {
    System.out.println("Refilling the " + color + " fountain pen.");
  }
}

class UseAndThrowPen extends FountainPen {
```

```
    public UseAndThrowPen(String color, String inkType, String nibSize) {
        super(color, inkType, nibSize);
    }

    public void write() {
        System.out.println("Writing with a " + color + " use-and-throw pen using " + inkType + " ink with a " + nibSize + " nib.");
    }

    public void dispose() {
        System.out.println("Disposing of the " + color + " use-and-throw pen.");
    }
}

public class PenTest {
    public static void main(String[] args) {
        Pen pen = new Pen("Blue", "gel");
        pen.write();

        FountainPen fountainPen = new FountainPen("Black", "liquid", "fine");
        fountainPen.write();
        fountainPen.refill();

        UseAndThrowPen useAndThrowPen = new UseAndThrowPen("Red", "ballpoint", "medium");
        useAndThrowPen.write();
        useAndThrowPen.dispose();
    }
}
```

OUTPUT:

```
C:\Users\yogir\OneDrive\Documents\Desktop\practice>java PenTest
Writing with a Blue pen using gel ink.
Writing with a Black fountain pen using liquid ink with a fine nib.
Refilling the Black fountain pen.
Writing with a Red use-and-throw pen using ballpoint ink with a medium nib.
Disposing of the Red use-and-throw pen.
```

2)CODE:

```
class RestaurantItem {
   String name;
   double price;

   public RestaurantItem(String name, double price) {
      this.name = name;
      this.price = price;
   }

   public void display() {
      System.out.println("Item: " + name + ", Price: ₹" + price);
   }
}

class Starters extends RestaurantItem {
   public Starters(String name, double price) {
      super(name, price);
   }

   public void display() {
      System.out.println("----------###----------");
      System.out.println("Starter: " + name + ", Price: ₹" + price);
      System.out.println("-----------------------");
   }

   public static void displayStarters() {
      Starters starter1 = new Starters("Garlic Bread", 120);
      Starters starter2 = new Starters("Bruschetta", 100);
      Starters starter3 = new Starters("Stuffed Mushrooms", 70);
      Starters starter4 = new Starters("Spring Rolls", 90);
      starter1.display();
      starter2.display();
```

```
        starter3.display();
        starter4.display();
    }
}

class MainItems extends RestaurantItem {
    public MainItems(String name, double price) {
        super(name, price);
    }

    public void display() {
        System.out.println("----------###----------");
        System.out.println("Main Item: " + name + ", Price: ₹" + price);
        System.out.println("----------");
    }

    public static void displayMainItems() {
        MainItems mainItem1 = new MainItems("Grilled Chicken",140);
        MainItems mainItem2 = new MainItems("Pasta Primavera", 210);
        MainItems mainItem3 = new MainItems("Beef Steak", 349);
        MainItems mainItem4 = new MainItems("Vegetable Curry", 100);
        mainItem1.display();
        mainItem2.display();
        mainItem3.display();
        mainItem4.display();
    }
}

class Desserts extends RestaurantItem {
    public Desserts(String name, double price) {
        super(name, price);
    }

    public void display() {
```

```
        System.out.println("----------###----------");
        System.out.println("Dessert: " + name + ", Price: ₹" + price);
         System.out.println("------------------------");

    }

    public static void displayDesserts() {
        Desserts dessert1 = new Desserts("Chocolate Cake", 60);
        Desserts dessert2 = new Desserts("Ice Cream Sundae", 80);
        Desserts dessert3 = new Desserts("Fruit Tart", 30);
        Desserts dessert4 = new Desserts("Cheesecake", 60);
        dessert1.display();
        dessert2.display();
        dessert3.display();
        dessert4.display();
    }
}

class Drinks extends RestaurantItem {
    public Drinks(String name, double price) {
        super(name, price);
    }

    public void display() {
        System.out.println("----------###----------");
        System.out.println("Drink: " + name + ", Price: ₹" + price);
        System.out.println("------------------------");
}

    public static void displayDrinks() {
        Drinks drink1 = new Drinks("Coke", 76);
        Drinks drink2 = new Drinks("Lemonade", 89);
        Drinks drink3 = new Drinks("Iced Tea", 200);
        Drinks drink4 = new Drinks("Water", 20);
```

```
        drink1.display();
        drink2.display();
        drink3.display();
        drink4.display();
    }
}

public class RestaurantMenu {
    public static void main(String[] args) {
        Starters.displayStarters();
        MainItems.displayMainItems();
        Desserts.displayDesserts();
        Drinks.displayDrinks();
    }
}
```

OUTPUT:

HYBRID INHERITANCE

1)Code:

```
class Bank {
   String bankName;

   public Bank(String bankName) {
      this.bankName = bankName;
   }

   public void displayBankInfo() {
      System.out.println("Bank Name: " + bankName);
```

```
    }
}

class SavingsAccount extends Bank {
    double interestRate;

    public SavingsAccount(String bankName, double interestRate) {
        super(bankName);
        this.interestRate = interestRate;
    }

    public void displaySavingsInfo() {
        System.out.println("Savings Account at " + bankName + " with Interest Rate: " + interestRate + "%");
    }
}

class CurrentAccount extends Bank {
    double overdraftLimit;

    public CurrentAccount(String bankName, double overdraftLimit) {
        super(bankName);
        this.overdraftLimit = overdraftLimit;
    }

    public void displayCurrentInfo() {
        System.out.println("Current Account at " + bankName + " with Overdraft Limit: ₹" + overdraftLimit);
    }
}

class LoanAccount extends SavingsAccount {
    double loanAmount;

    public LoanAccount(String bankName, double interestRate, double loanAmount) {
        super(bankName, interestRate);
```

```
        this.loanAmount = loanAmount;
    }

    public void displayLoanInfo() {
        System.out.println("Loan Account at " + bankName + " with Loan Amount: ₹" + loanAmount + " and Interest Rate: " + interestRate + "%");
    }
}

public class BankTest {
    public static void main(String[] args) {
        SavingsAccount savings = new SavingsAccount("SBI Bank", 4.5);
        savings.displayBankInfo();
        savings.displaySavingsInfo();

        CurrentAccount current = new CurrentAccount("HDFC Bank", 5000);
        current.displayBankInfo();
        current.displayCurrentInfo();

        LoanAccount loan = new LoanAccount("icici Bank", 5.0, 200000);
        loan.displayBankInfo();
        loan.displaySavingsInfo();
        loan.displayLoanInfo();
    }
}
```

OUTPUT:

```
C:\path\to\your\java\files> java BankTest
Bank Name: SBI
Savings Account at SBI with Interest Rate: 4.5%
Bank Name: HDFC
Current Account at HDFC with Overdraft Limit: ₹5000.0
Bank Name: ICICI
Savings Account at ICICI with Interest Rate: 5.0%
Loan Account at ICICI with Loan Amount: ₹200000.0 and Interest
```

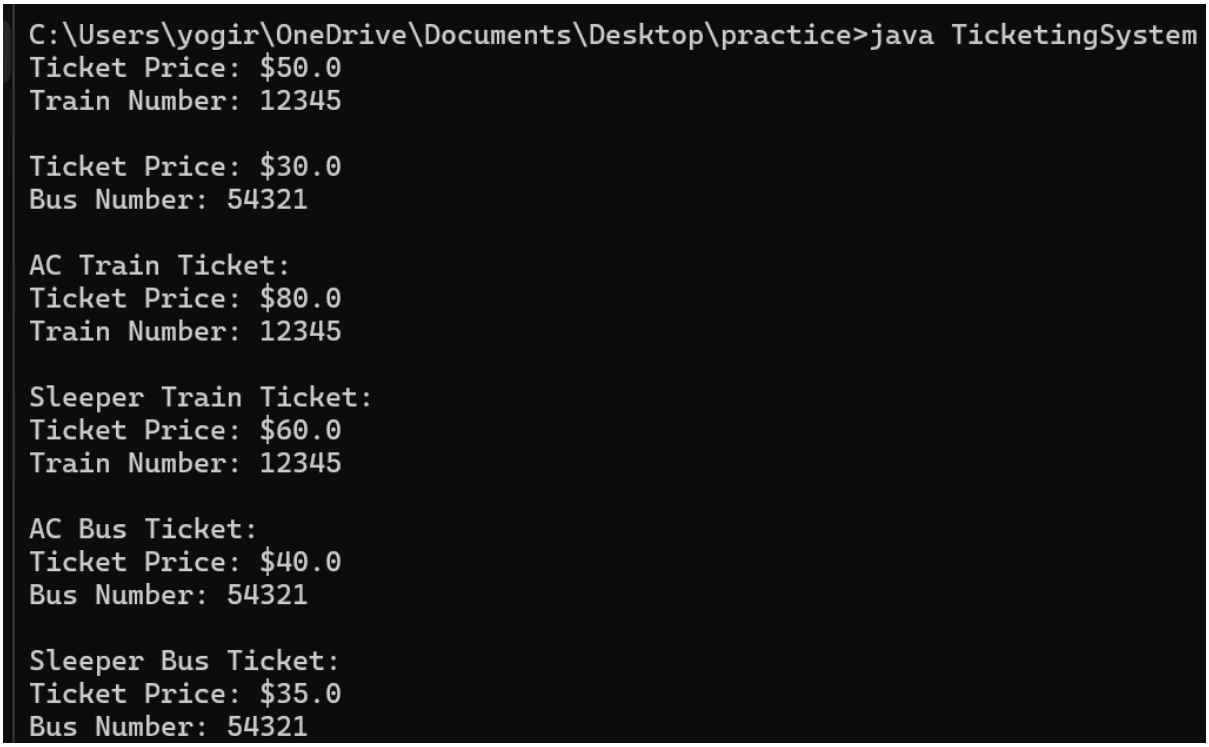
2)CODE:

```
class Ticket {
  protected double price;

  public Ticket(double price) {
    this.price = price;
  }

  public double getPrice() {
    return price;
  }

  public void displayTicketInfo() {
    System.out.println("Ticket Price: $" + price);
  }
}

class TrainTicket extends Ticket {
  private String trainNumber;

  public TrainTicket(double price, String trainNumber) {
    super(price);
    this.trainNumber = trainNumber;
  }

  public void displayTicketInfo() {
    super.displayTicketInfo();
    System.out.println("Train Number: " + trainNumber);
  }
}

class BusTicket extends Ticket {
  private String busNumber;

  public BusTicket(double price, String busNumber) {
    super(price);
    this.busNumber = busNumber;
  }

  public void displayTicketInfo() {
    super.displayTicketInfo();
    System.out.println("Bus Number: " + busNumber);
  }
}

class ACTicket extends TrainTicket {
  public ACTicket(double price, String trainNumber) {
    super(price, trainNumber);
  }

  public void displayTicketInfo() {
    System.out.println("AC Train Ticket:");
    super.displayTicketInfo();
```

```
  }
}

class SleeperTicket extends TrainTicket {
  public SleeperTicket(double price, String trainNumber) {
    super(price, trainNumber);
  }

  public void displayTicketInfo() {
    System.out.println("Sleeper Train Ticket:");
    super.displayTicketInfo();
  }
}

class ACBusTicket extends BusTicket {
  public ACBusTicket(double price, String busNumber) {
    super(price, busNumber);
  }

  public void displayTicketInfo() {
    System.out.println("AC Bus Ticket:");
    super.displayTicketInfo();
  }
}

class SleeperBusTicket extends BusTicket {
  public SleeperBusTicket(double price, String busNumber) {
    super(price, busNumber);
  }

  public void displayTicketInfo() {
    System.out.println("Sleeper Bus Ticket:");
    super.displayTicketInfo();
  }
}

public class TicketingSystem {
  public static void main(String[] args) {
    Ticket trainTicket = new TrainTicket(50.0, "12345");
    Ticket busTicket = new BusTicket(30.0, "54321");
    Ticket acTrainTicket = new ACTicket(80.0, "12345");
    Ticket sleeperTrainTicket = new SleeperTicket(60.0, "12345");
    Ticket acBusTicket = new ACBusTicket(40.0, "54321");
    Ticket sleeperBusTicket = new SleeperBusTicket(35.0, "54321");

    trainTicket.displayTicketInfo();
    System.out.println();
    busTicket.displayTicketInfo();
    System.out.println();
    acTrainTicket.displayTicketInfo();
    System.out.println();
    sleeperTrainTicket.displayTicketInfo();
    System.out.println();
    acBusTicket.displayTicketInfo();
```

```
        System.out.println();
        sleeperBusTicket.displayTicketInfo();
    }
}
```

OUTPUT:



POLYMORPHISM

Constructor programs

1)CODE:

```
class Shape { private String color;

public Shape(String color) {
    this.color = color;
    System.out.println("A shape of color " + color + " has been created.");
}

public void displayColor() {
    System.out.println("Color: " + color);
}


}

class Circle extends Shape { private double radius;
```

```
public Circle(String color, double radius) {
    super(color);
    this.radius = radius;
    System.out.println("Circle created with radius: " + radius);
}

public void area() {
    double area = Math.PI * radius * radius;
    System.out.println("Area of Circle: " + area);
}


}
```

class Rectangle extends Shape { private double length; private double width;

```
public Rectangle(String color, double length, double width) {
    super(color);
    this.length = length;
    this.width = width;
    System.out.println("Rectangle created with length: " + length + " and width: " +
width);
}

public void area() {
    double area = length * width;
    System.out.println("Area of Rectangle: " + area);
}}
```

public class ShapeTest { public static void main(String[] args) { Circle circle = new Circle("Red", 5.0); circle.area(); circle.displayColor();

```
   Rectangle rectangle = new Rectangle("Blue", 4.0, 6.0);
    rectangle.area();
    rectangle.displayColor();
}
}
```

OUTPUT:

```
C:\Users\yogir\OneDrive\Documents\Desktop\practice>java ShapeTest
A shape of color Red has been created.
Circle created with radius: 5.0
Area of Circle: 78.53981633974483
Color: Red
A shape of color Blue has been created.
Rectangle created with length: 4.0 and width: 6.0
Area of Rectangle: 24.0
Color: Blue
```

CONSTRUCTOR OVERLOADING

1)CODE:

class Payment {

```
    private String paymentId;
    private double amount;
    private String paymentMethod;
    private String currency;

    public Payment(String paymentId, double amount, String paymentMethod) {
        this.paymentId = paymentId;
        this.amount = amount;
        this.paymentMethod = paymentMethod;
        this.currency = "USD";
    }

    public Payment(String paymentId, double amount, String paymentMethod, String currency) {
        this.paymentId = paymentId;
        this.amount = amount;
        this.paymentMethod = paymentMethod;
        this.currency = currency;
    }

    public Payment(String paymentId, double amount) {
        this.paymentId = paymentId;
        this.amount = amount;
        this.paymentMethod = "Cash";
        this.currency = "USD";
    }

    public void displayPaymentDetails() {
        System.out.println("Payment ID: " + paymentId);
        System.out.println("Amount: " + amount);
        System.out.println("Payment Method: " + paymentMethod);
        System.out.println("Currency: " + currency);
        System.out.println("----------------------------");
    }
}
```

```
public class OnlinePaymentSystem {
  public static void main(String[] args) {
    Payment creditCardPayment = new Payment("CC123", 150.00, "Credit Card");
    creditCardPayment.displayPaymentDetails();

    Payment bankTransferPayment = new Payment("BT456", 200.00, "Bank Transfer", "EUR");
    bankTransferPayment.displayPaymentDetails();

    Payment cashPayment = new Payment("C789", 50.00);
    cashPayment.displayPaymentDetails();
  }
}
```

OUTPUT:

```
C:\Users\yogir\OneDrive\Documents\Desktop\practice>java OnlinePaymentSystem
Payment ID: CC123
Amount: 150.0
Payment Method: Credit Card
Currency: USD
--------------------------
Payment ID: BT456
Amount: 200.0
Payment Method: Bank Transfer
Currency: EUR
--------------------------
Payment ID: C789
Amount: 50.0
Payment Method: Cash
Currency: USD
--------------------------
```

Method Overloading

1)CODE:

```
class Shop {
  public void calculateTotalPrice(double price, int quantity) {
    double total = price * quantity;
    System.out.println("Total Price (without discount): " + total);
  }

  public void calculateTotalPrice(double price, int quantity, double discount) {
    double total = price * quantity;
```

```
        total=total- (total * discount / 100);
        System.out.println("Total Price (with discount): " + total);
    }

    public void calculateTotalPrice(double price, int quantity, double discount, double tax) {
        double total = price * quantity;
        total=total- (total * discount / 100);
        total=total+ (total * tax / 100);
        System.out.println("Total Price (with discount and tax): " + total);
    }
}

public class ShopLite {
    public static void main(String[] args) {
        Shop shop = new Shop();

        double price1 = 100.0;
        int quantity1 = 2;
        double discount1 = 10.0;
        double tax1 = 5.0;

        shop.calculateTotalPrice(price1, quantity1);
        shop.calculateTotalPrice(price1, quantity1, discount1);
        shop.calculateTotalPrice(price1, quantity1, discount1, tax1);
    }
}
```

OUTPUT:

```
C:\Users\yogir\OneDrive\Documents\Desktop\practice>java ShopLite
Total Price (without discount): 200.0
Total Price (with discount): 180.0
Total Price (with discount and tax): 189.0
```

2)CODE:

class Multiplier { public void multiply(int a, int b) { int result = a * b; System.out.println("Multiplication of two numbers: " + result); }

```
public void multiply(int a, int b, int c) {
    int result = a * b * c;
    System.out.println("Multiplication of three numbers: " + result);
}

public void multiply(int a, int b, int c, int d) {
    int result = a * b * c * d;
    System.out.println("Multiplication of four numbers: " + result);
}


}
```

public class MultiplierExample { public static void main(String[] args) { Multiplier multiplier = new Multiplier();

```
   multiplier.multiply(2, 3);
    multiplier.multiply(2, 3, 4);
    multiplier.multiply(2, 3, 4, 5);
}


}
```

OUTPUT:

```
C:\Users\yogir\OneDrive\Documents\Desktop\practice>java MultiplierExample
Multiplication of two numbers: 6
Multiplication of three numbers: 24
Multiplication of four numbers: 120
```

METHOD OVERIDING

1)CODE:

class Home {

public void display() {

System.out.println("We are in the home ");

}

}

class Apartment extends Home { public void display() System.out.println("We are in the apartment ") }}class Company extends Home { public void display() System.out.println("We are in the Company"); }}public class Room { public static void main(String[] args) Home myHome = new Home() Home myApartment = new Apartment();

Company myObj=new Company();

myHome.display();

```
    myApartment.display();
    myObj.display();
  }}
```

OUTPUT:

2)CODE:

```
class DeliveryItem {
  private double basePrice;
  public void setBasePrice(double basePrice) {
    this.basePrice = basePrice;
  }
  public double getBasePrice() {
    return basePrice;
  }
  public void calculatePrice() {
    System.out.println("Price of Regular Delivery Item: " + basePrice);
  }
}
class ExpressDeliveryItem extends DeliveryItem {
  private double expressFee;
  public void setExpressFee(double expressFee) {
    this.expressFee = expressFee;
```

```
    }

    public void calculatePrice() {

        double totalPrice = getBasePrice() + expressFee;

        System.out.println("Price of Express Delivery Item: " + totalPrice);

    }

}

public class Delivery {

    public static void main(String[] args) {

        DeliveryItem regularItem = new DeliveryItem();



       regularItem.setBasePrice(100.0);

        ExpressDeliveryItem expressItem = new ExpressDeliveryItem();



  expressItem.setBasePrice(100.0);

        expressItem.setExpressFee(20.0);

        regularItem.calculatePrice();

        expressItem.calculatePrice();

    }

}
```

OUTPUT:

```
C:\Users\yogir\OneDrive\Documents\Desktop\practice>java Delivery
Price of Regular Delivery Item: 100.0
Price of Express Delivery Item: 120.0
```

ABSTRACTION

INTERFACE PROGRAMS

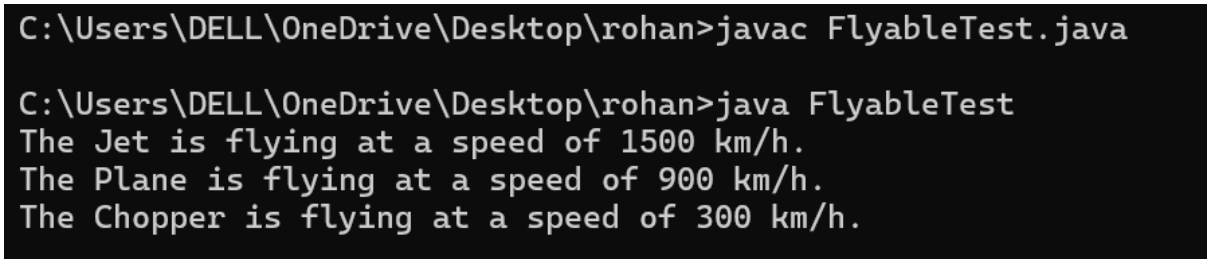
1)CODE:

```
interface Flyable {
```

```
    void fly_obj();
}

class Jet implements Flyable {
    public void fly_obj() {
        System.out.println("The Jet is flying at a speed of 1500 km/h.");
    }
}

class Plane implements Flyable {
    public void fly_obj() {
        System.out.println("The Plane is flying at a speed of 900 km/h.");
    }
}

class Chopper implements Flyable {
    public void fly_obj() {
        System.out.println("The Chopper is flying at a speed of 300 km/h.");
    }
}

public class FlyableTest {
    public static void main(String[] args) {
        Flyable jet = new Jet();
        Flyable plane = new Plane();
        Flyable chopper = new Chopper();

        jet.fly_obj();
        plane.fly_obj();
```

```
    chopper.fly_obj();
  }
}
```

OUTPUT:

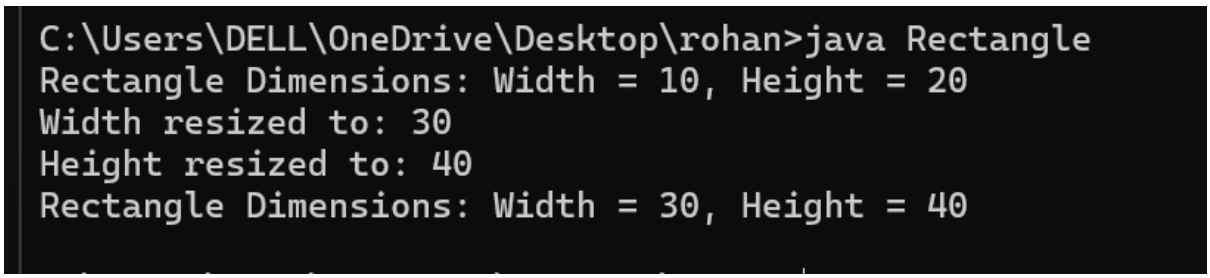


2)CODE:

```
interface Resizable {
  void resizeWidth(int width);
  void resizeHeight(int height);
}

class Rectangle implements Resizable {
  private int width;
  private int height;

  public Rectangle(int width, int height) {
    this.width = width;
    this.height = height;
  }

  public void resizeWidth(int width) {
    if (width > 0) {
      this.width = width;
      System.out.println("Width resized to: " + this.width);
    } else {
```

```
        System.out.println("Invalid width value");
    }
  }

  public void resizeHeight(int height) {
    if (height > 0) {
      this.height = height;
      System.out.println("Height resized to: " + this.height);
    } else {
      System.out.println("Invalid height value");
    }
  }

  public void display() {
    System.out.println("Rectangle Dimensions: Width = " + width + ", Height = " + height);
  }

  public static void main(String[] args) {
    Rectangle rect = new Rectangle(10, 20);
    rect.display();
    rect.resizeWidth(30);
    rect.resizeHeight(40);
    rect.display();
  }}
```

OUTPUT:

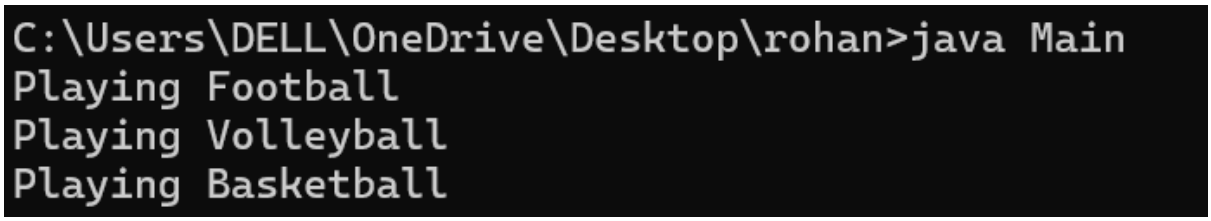


3)CODE:

```
interface Playable {
   void play();
}

class Football implements Playable {
   public void play() {
      System.out.println("Playing Football");
   }
}

class Volleyball implements Playable {
   public void play() {
      System.out.println("Playing Volleyball");
   }
}

class Basketball implements Playable {
   public void play() {
      System.out.println("Playing Basketball");
   }
}

public class Main {
   public static void main(String[] args) {
      Playable football = new Football();
      Playable volleyball = new Volleyball();
      Playable basketball = new Basketball();

      football.play();
```

```
        volleyball.play();
        basketball.play();
    }
}
```

OUTPUT:



4)CODE:

```
interface Playable {
    void play();
}

class Football implements Playable {
    public void play() {
        System.out.println("Playing Football");
    }
}

class Volleyball implements Playable {
    public void play() {
        System.out.println("Playing Volleyball");
    }
}

class Basketball implements Playable {
    public void play() {
        System.out.println("Playing Basketball");
    }
```

```
}


public class Main {
  public static void main(String[] args) {
    Playable football = new Football();
    Playable volleyball = new Volleyball();
    Playable basketball = new Basketball();

    football.play();
    volleyball.play();
    basketball.play();
  }
}
```

OUTPUT:

```
C:\Users\DELL\OneDrive\Desktop\rohan>java Main
Car order placed.
Tracking car order.
Car order canceled.
```

ABSTRACT CLASSES PROGRAMS

1)CODE:

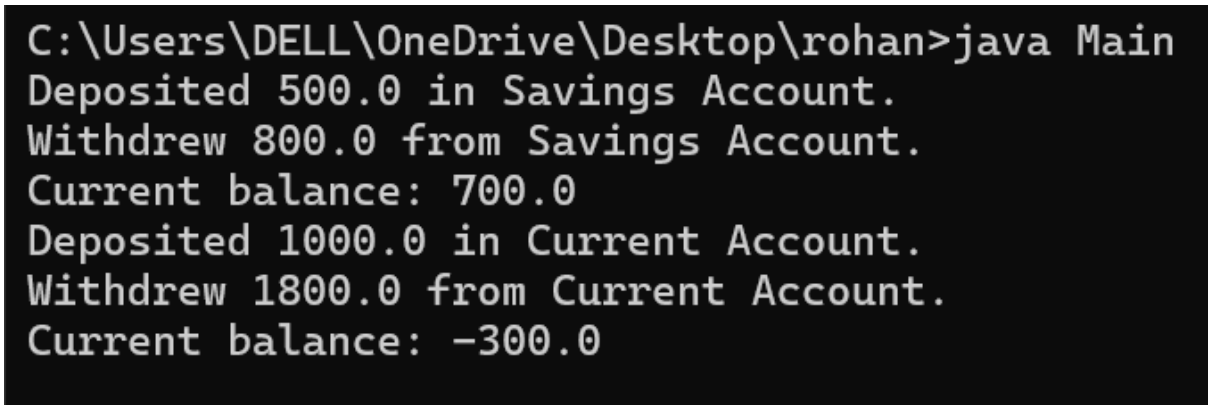
```
abstract class BankAccount {
  double balance;

  public BankAccount(double balance) {
    this.balance = balance;
  }

  abstract void deposit(double amount);
  abstract void withdraw(double amount);
```

```
    public void displayBalance() {
        System.out.println("Current balance: " + balance);
    }
}

class SavingsAccount extends BankAccount {
    private static final double MIN_BALANCE = 500;

    public SavingsAccount(double balance) {
        super(balance);
    }

    public void deposit(double amount) {
        balance += amount;
        System.out.println("Deposited " + amount + " in Savings Account.");
    }

    public void withdraw(double amount) {
        if (balance - amount >= MIN_BALANCE) {
            balance -= amount;
            System.out.println("Withdrew " + amount + " from Savings Account.");
        } else {
            System.out.println("Insufficient balance. Minimum balance must be maintained.");
        }
    }
}

class CurrentAccount extends BankAccount {
    private static final double OVERDRAFT_LIMIT = 1000;
```

```
    public CurrentAccount(double balance) {
        super(balance);
    }

    public void deposit(double amount) {
        balance += amount;
        System.out.println("Deposited " + amount + " in Current Account.");
    }

    public void withdraw(double amount) {
        if (balance - amount >= -OVERDRAFT_LIMIT) {
            balance -= amount;
            System.out.println("Withdrew " + amount + " from Current Account.");
        } else {
            System.out.println("Overdraft limit exceeded. Cannot withdraw.");
        }
    }
}

public class Main {
    public static void main(String[] args) {
        BankAccount savings = new SavingsAccount(1000);
        savings.deposit(500);
        savings.withdraw(800);
        savings.displayBalance();

        BankAccount current = new CurrentAccount(500);
        current.deposit(1000);
        current.withdraw(1800);
```

```
    current.displayBalance();
  }
}
```

OUTPUT:

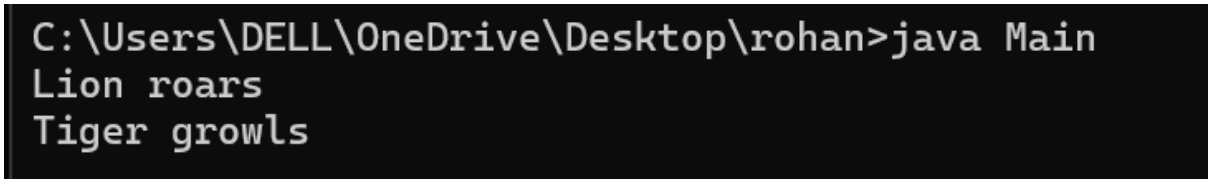


2)CODE:

```
abstract class Animal {
  abstract void sound();
}

class Lion extends Animal {
  public void sound() {
    System.out.println("Lion roars");
  }
}

class Tiger extends Animal {
  public void sound() {
    System.out.println("Tiger growls");
  }
}

public class Main {
```

```
    public static void main(String[] args) {
        Animal lion = new Lion();
        Animal tiger = new Tiger();

        lion.sound();
        tiger.sound();
    }
}
```

OUTPUT:

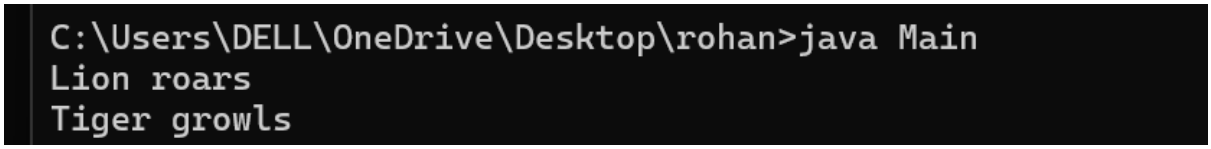


3)CODE:

```
abstract class Animal {
    abstract void sound();
}

class Lion extends Animal {
    public void sound() {
        System.out.println("Lion roars");
    }
}

class Tiger extends Animal {
    public void sound() {
        System.out.println("Tiger growls");
    }
}

abstract class RapidoBooking {
```

```
    String pickupLocation;
    String dropLocation;

    public RapidoBooking(String pickupLocation, String dropLocation) {
        this.pickupLocation = pickupLocation;
        this.dropLocation = dropLocation;
    }

    abstract void bookRide();
}

class BikeRide extends RapidoBooking {
    public BikeRide(String pickupLocation, String dropLocation) {
        super(pickupLocation, dropLocation);
    }

    public void bookRide() {
        System.out.println("Bike ride booked from " + pickupLocation + " to " + dropLocation);
    }
}



public class Main {
    public static void main(String[] args) {
        Animal lion = new Lion();
        Animal tiger = new Tiger();

        lion.sound();
        tiger.sound();
    }}
```

OUTPUT:

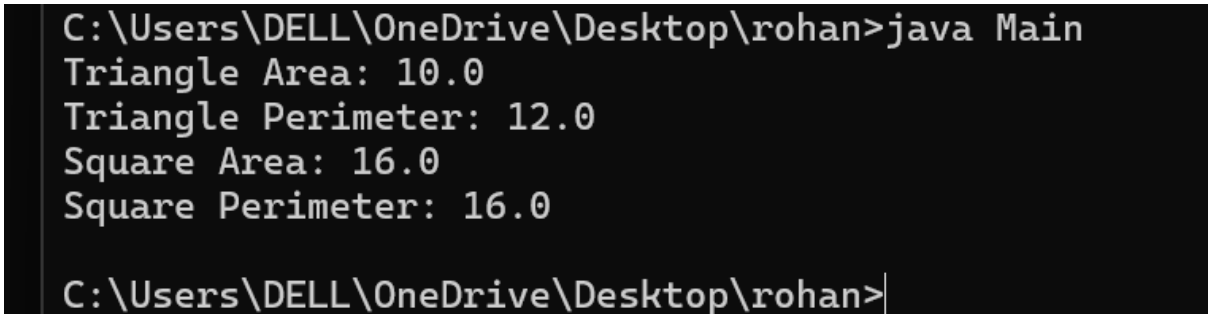


4)CODE:

```
abstract class GeometricShape {
   abstract void area();
   abstract void perimeter();
}

class Triangle extends GeometricShape {
   private double base, height, side1, side2, side3;

   public Triangle(double base, double height, double side1, double side2, double side3) {
      this.base = base;
      this.height = height;
      this.side1 = side1;
      this.side2 = side2;
      this.side3 = side3;
   }

   public void area() {
      System.out.println("Triangle Area: " + (0.5 * base * height));
   }

   public void perimeter() {
      System.out.println("Triangle Perimeter: " + (side1 + side2 + side3));
   }
}
```

```
class Square extends GeometricShape {
    private double side;

    public Square(double side) {
        this.side = side;
    }

    public void area() {
        System.out.println("Square Area: " + (side * side));
    }

    public void perimeter() {
        System.out.println("Square Perimeter: " + (4 * side));
    }
}

public class Main {
    public static void main(String[] args) {
        GeometricShape triangle = new Triangle(5, 4, 3, 4, 5);
        GeometricShape square = new Square(4);

        triangle.area();
        triangle.perimeter();

        square.area();
        square.perimeter();
    }
}
```

OUTPUT:



ENCAPSULATION

ENCAPSULATION PROGRAMS

1)CODE:

```
class Person {

  private String name;

  private int age;

  private String country;



  public void getName() {

    System.out.println("Name: " + name);

  }



  public void setName(String name) {

    this.name = name;

  }



  public void getAge() {

    System.out.println("Age: " + age);

  }



  public void setAge(int age) {

    this.age = age;
```

```
    }

    public void getCountry() {
        System.out.println("Country: " + country);
    }

    public void setCountry(String country) {
        this.country = country;
    }
}

public class Main {
    public static void main(String[] args) {
        Person person = new Person();
        person.setName("John");
        person.setAge(30);
        person.setCountry("USA");

        person.getName();
        person.getAge();
        person.getCountry();
    }
}
```

OUTPUT:

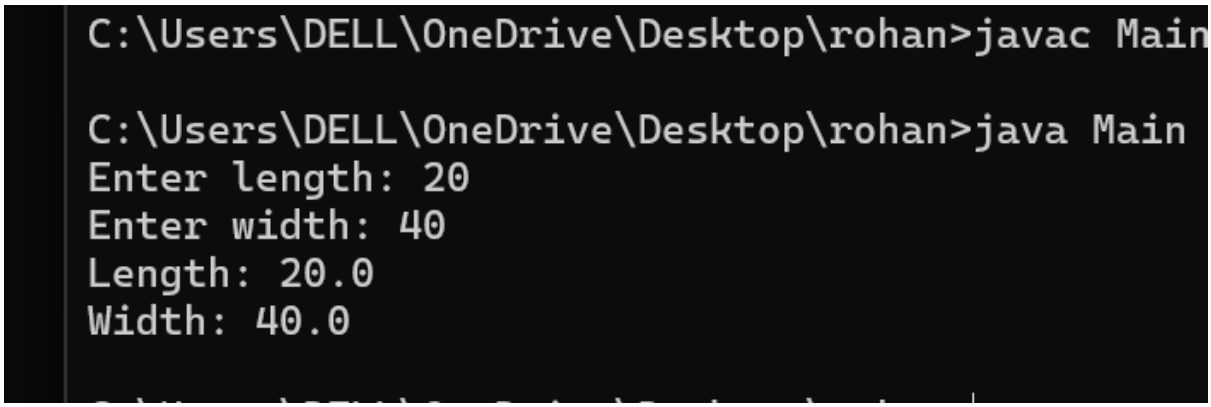
```
C:\Users\DELL\OneDrive\Desktop\rohan>java Main
Name: John
Age: 30
Country: USA

C:\Users\DELL\OneDrive\Desktop\rohan>
```

2)CODE:

```
import java.util.Scanner;

class Rectangle {
  private double length;
  private double width;

  public void setLength(double length) {
    this.length = length;
  }

  public void setWidth(double width) {
    this.width = width;
  }

  public void getLength() {
    System.out.println("Length: " + length);
  }

  public void getWidth() {
    System.out.println("Width: " + width);
  }
}

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

    System.out.print("Enter length: ");
    double length = sc.nextDouble();
```

```
        rectangle.setLength(length);


        System.out.print("Enter width: ");
        double width = sc.nextDouble();
        rectangle.setWidth(width);


        rectangle.getLength();
        rectangle.getWidth();


        sc.close();
    }
}
```

OUTPUT:



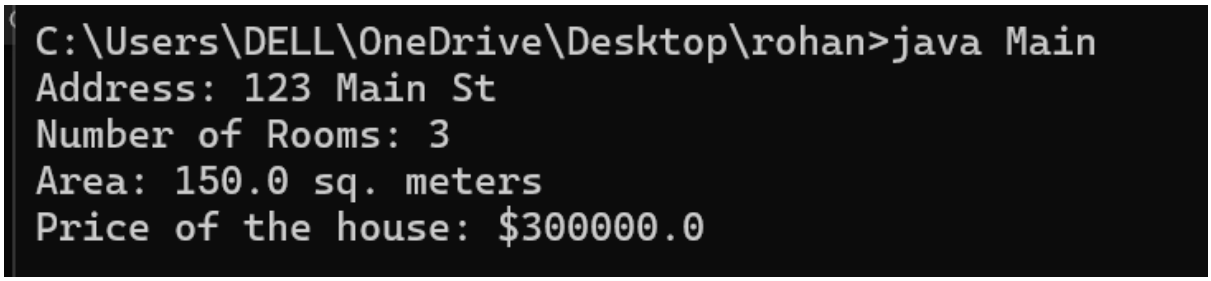
3)CODE:

```
class House {
    private String address;
    private int numberOfRooms;
    private double area;


    public House(String address, int numberOfRooms, double area) {
        this.address = address;
```

```
    this.numberOfRooms = numberOfRooms;
    this.area = area;
}

public String getAddress() {
    return address;
}

public void setAddress(String address) {
    this.address = address;
}

public int getNumberOfRooms() {
    return numberOfRooms;
}

public void setNumberOfRooms(int numberOfRooms) {
    this.numberOfRooms = numberOfRooms;
}

public double getArea() {
    return area;
}

public void setArea(double area) {
    this.area = area;
}

public double calculatePrice(double pricePerSquareMeter) {
    return area * pricePerSquareMeter;
```

```
    }
}

public class Main {
    public static void main(String[] args) {
        House house = new House("123 Main St", 3, 150.0);
        System.out.println("Address: " + house.getAddress());
        System.out.println("Number of Rooms: " + house.getNumberOfRooms());
        System.out.println("Area: " + house.getArea() + " sq. meters");
        double pricePerSquareMeter = 2000.0;
        double price = house.calculatePrice(pricePerSquareMeter);
        System.out.println("Price of the house: $" + price);
    }
}
```

OUTPUT:



4)CODE:

```
class CricketPlayer {
    private String name;
    private int runs;
    private int wickets;

    public void setName(String name) {
        this.name = name;
    }
```

```
    public void setRuns(int runs) {
        this.runs = runs;
    }

    public void setWickets(int wickets) {
        this.wickets = wickets;
    }

    public void getName() {
        System.out.println("Player Name: " + name);
    }

    public void getRuns() {
        System.out.println("Total Runs: " + runs);
    }

    public void getWickets() {
        System.out.println("Total Wickets: " + wickets);
    }
}

public class Main {
    public static void main(String[] args) {
        CricketPlayer player = new CricketPlayer();
        player.setName("Virat Kohli");
        player.setRuns(12000);
        player.setWickets(4);

        player.getName();
```

```
    player.getRuns();
    player.getWickets();
  }
}
```

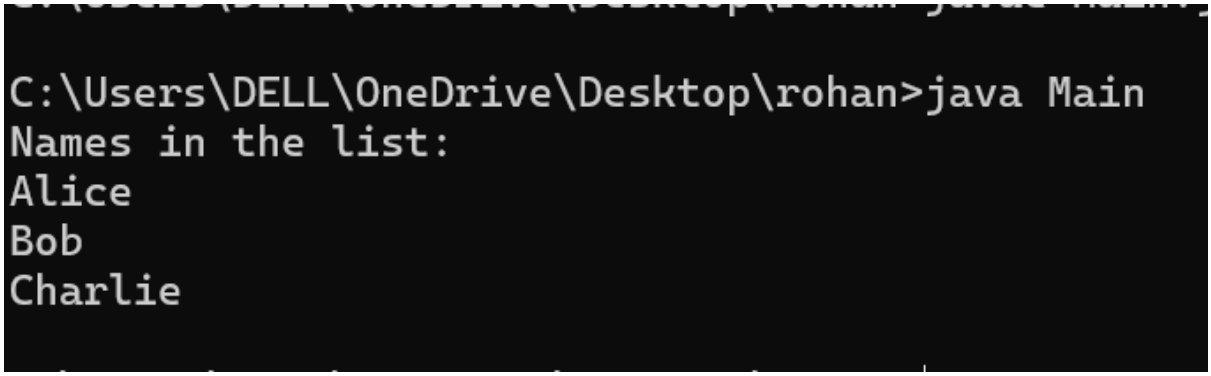
CODE:

PACKAGES PROGRAMS

1)CODE:

```
import java.util.ArrayList;

public class Main {
  public static void main(String[] args) {
    ArrayList<String> names = new ArrayList<>();
    names.add("Alice");
    names.add("Bob");
    names.add("Charlie");

    System.out.println("Names in the list:");
    for (String name : names) {
      System.out.println(name);
    }
  }
}
```

OUTPUT:



2)CODE:

```
import java.io.*;

import java.net.*;



public class Main {

   public static void main(String[] args) {

      String hostname = "localhost";

      int port = 12345;



      try (Socket socket = new Socket(hostname, port);

         PrintWriter out = new PrintWriter(socket.getOutputStream(), true);

         BufferedReader in = new BufferedReader(new InputStreamReader(socket.getInputStream()))) {



         out.println("Hello Server!");

         String response = in.readLine();

         System.out.println("Server response: " + response);

      } catch (IOException e) {

         e.printStackTrace();

      }

  }}
```

OUTPUT:

```
C:\Users\DELL\OneDrive\Desktop\rohan>java Main
java.net.ConnectException: Connection refused: connect
        at java.base/sun.nio.ch.Net.connect0(Native Method)
        at java.base/sun.nio.ch.Net.connect(Net.java:589)
        at java.base/sun.nio.ch.Net.connect(Net.java:578)
        at java.base/sun.nio.ch.NioSocketImpl.connect(NioSocketImpl.java:583)
        at java.base/java.net.SocksSocketImpl.connect(SocksSocketImpl.java:327)
        at java.base/java.net.Socket.connect(Socket.java:760)
        at java.base/java.net.Socket.connect(Socket.java:695)
        at java.base/java.net.Socket.<init>(Socket.java:564)
        at java.base/java.net.Socket.<init>(Socket.java:328)
        at Main.main(Main.java:9)
```

3)CODE:

```
import java.time.*;

import java.time.format.DateTimeFormatter;



public class Main {

   public static void main(String[] args) {

      // Get the current date, time, and date-time

      LocalDate currentDate = LocalDate.now();

      LocalTime currentTime = LocalTime.now();

      LocalDateTime currentDateTime = LocalDateTime.now();



      // Format the current date-time

      DateTimeFormatter formatter = DateTimeFormatter.ofPattern("yyyy-MM-dd HH:mm:ss");

      String formattedDateTime = currentDateTime.format(formatter);



      // Get the current date and time in New York timezone

      ZonedDateTime zonedDateTime = ZonedDateTime.now(ZoneId.of("America/New_York"));



      // Define a past date

      LocalDate pastDate = LocalDate.of(2020, Month.JANUARY, 1);
```

```
        // Calculate the period between the past date and the current date
        Period period = Period.between(pastDate, currentDate);

        // Print the results
        System.out.printf("Current Date: %s%n", currentDate);
        System.out.printf("Current Time: %s%n", currentTime);
        System.out.printf("Formatted DateTime: %s%n", formattedDateTime);
        System.out.printf("Zoned DateTime (New York): %s%n", zonedDateTime);
        System.out.printf("Period from 2020-01-01 to now: %d years, %d months, %d days%n",
                period.getYears(), period.getMonths(), period.getDays());
    }
}
```

OUTPUT:

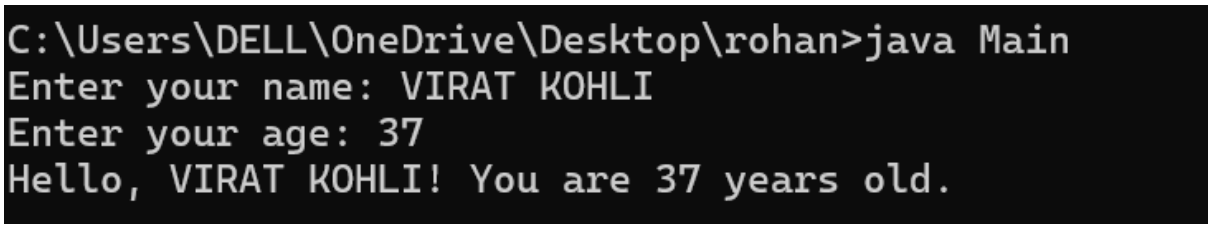
```
C:\Users\DELL\OneDrive\Desktop\rohan>javac Main.java

C:\Users\DELL\OneDrive\Desktop\rohan>java Main
Current Date: 2025-04-03
Current Time: 23:11:12.310761500
Formatted DateTime: 2025-04-03 23:11:12
Zoned DateTime (New York): 2025-04-03T13:41:12.319745500-04:00[America/New_York]
Period from 2020-01-01 to now: 5 years, 3 months, 2 days
```

4)CODE:

```
import java.util.Scanner;

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

        System.out.print("Enter your name: ");
        String name = scanner.nextLine();

        System.out.print("Enter your age: ");
        int age = scanner.nextInt();
```

```
        System.out.println("Hello, " + name + "! You are " + age + " years old.");


        scanner.close();
    }
}
```

OUTPUT:



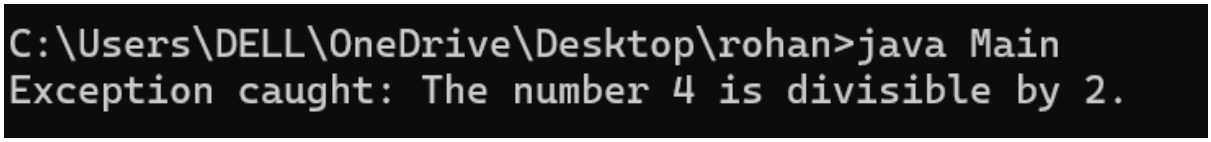
EXCEPTION HANDLING

1)CODE:

```
class EvenNumberException extends Exception {
    public EvenNumberException(String message) {
        super(message);
    }
}


public class Main {
    public static void main(String[] args) {
        try {
            checkNumber(4); // Change this number to test different cases
        } catch (EvenNumberException e) {
            System.out.println("Exception caught: " + e.getMessage());
        }
    }
```

```
  public static void checkNumber(int number) throws EvenNumberException {
    if (number % 2 == 0) {
      throw new EvenNumberException("The number " + number + " is divisible by 2.");
    } else {
      System.out.println("The number " + number + " is not divisible by 2.");
    }
  }
}
```

OUTPUT:

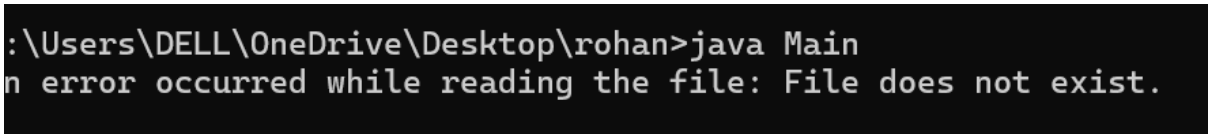


2)CODE:

```
import java.io.IOException;
import java.nio.file.Files;
import java.nio.file.Path;
import java.nio.file.Paths;
import java.util.List;

class NegativeNumberException extends Exception {
  public NegativeNumberException(String message) {
    super(message);
  }
}

public class Main {
  public static void main(String[] args) {
    String filePath = "numbers.txt"; // Change this to the path of your file
```

```
    try {
        checkNumbers(filePath);
        System.out.println("All numbers are non-negative.");
    } catch (NegativeNumberException e) {
        System.out.println("Exception caught: " + e.getMessage());
    } catch (IOException e) {
        System.out.println("An error occurred while reading the file: " + e.getMessage());
    }
}

public static void checkNumbers(String filePath) throws NegativeNumberException, IOException {
    Path path = Paths.get(filePath);

    if (Files.notExists(path)) {
        throw new IOException("File does not exist.");
    }

    List<String> lines = Files.readAllLines(path);

    for (String line : lines) {
        try {
            int number = Integer.parseInt(line.trim());
            if (number < 0) {
                throw new NegativeNumberException("Negative number found: " + number);
            }
        } catch (NumberFormatException e) {
            System.out.println("Invalid number format: " + line);
        }
    }
}}
```

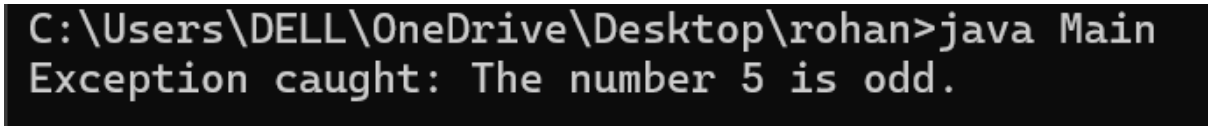
OUTPUT:



3)CODE:

```
class OddNumberException extends Exception {
  public OddNumberException(String message) {
    super(message);
  }
}

public class Main {
  public static void main(String[] args) {
    try {
      checkNumber(5); // Change this number to test different cases
    } catch (OddNumberException e) {
      System.out.println("Exception caught: " + e.getMessage());
    }
  }

  public static void checkNumber(int number) throws OddNumberException {
    if (number % 2 != 0) {
      throw new OddNumberException("The number " + number + " is odd.");
    } else {
      System.out.println("The number " + number + " is even.");
    }
  }
}
```

OUTPUT:

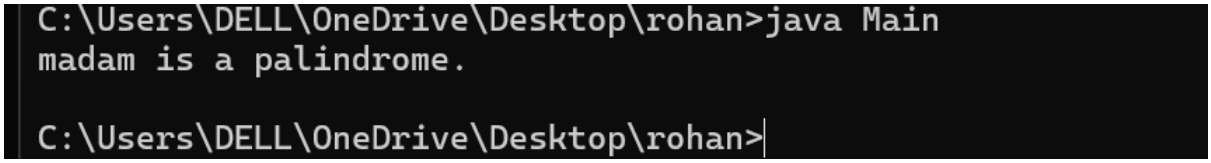


4)CODE:

```
class InvalidInputException extends Exception {
   public InvalidInputException(String message) {
      super(message);
   }
}


public class Main {
   public static void main(String[] args) {
      String input = "madam"; // Change this to test different cases


      try {
         checkPalindrome(input);
         System.out.println(input + " is a palindrome.");
      } catch (InvalidInputException e) {
         System.out.println("Exception caught: " + e.getMessage());
      }
   }


   public static void checkPalindrome(String str) throws InvalidInputException {
      if (str == null || str.isEmpty()) {
         throw new InvalidInputException("Input string is null or empty.");
      }


      String reversedStr = new StringBuilder(str).reverse().toString();
      if (!str.equals(reversedStr)) {
```

```
        throw new InvalidInputException(str + " is not a palindrome.");
      }
    }
}
```

OUTPUT:

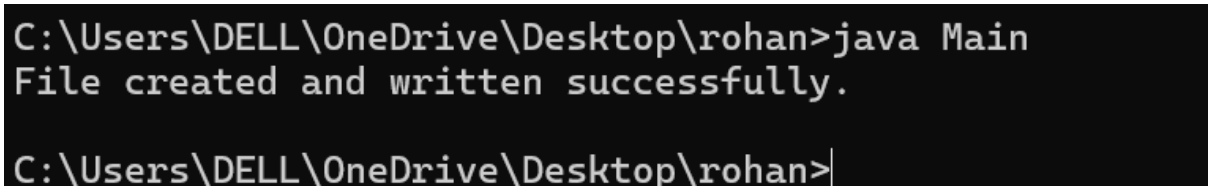


FILE HANDLING

1)CODE:

```
import java.io.FileWriter;
import java.io.IOException;

public class Main {
   public static void main(String[] args) {
      try (FileWriter writer = new FileWriter("example.txt")) {
         writer.write("Hello, this is a sample file.\n");
         writer.write("Java File Handling Example.\n");
         System.out.println("File created and written successfully.");
      } catch (IOException e) {
         System.out.println("An error occurred.");
         e.printStackTrace();
      }
   }
}
```

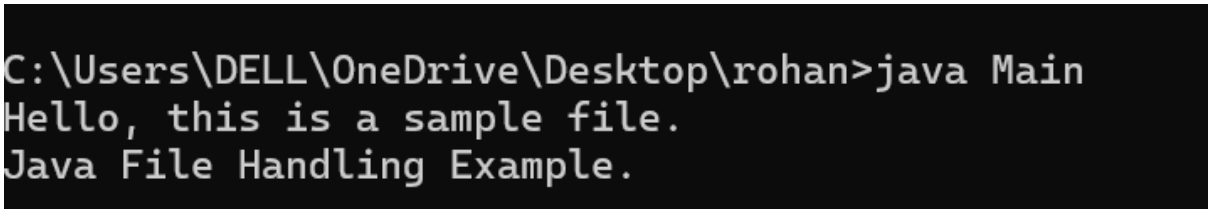
OUTPUT:



2)CODE:

```
import java.io.File;

import java.io.FileNotFoundException;

import java.util.Scanner;



public class Main {

  public static void main(String[] args) {

    try {

      File file = new File("example.txt");

      Scanner reader = new Scanner(file);

      while (reader.hasNextLine()) {

        System.out.println(reader.nextLine());

      }

      reader.close();

    } catch (FileNotFoundException e) {

      System.out.println("File not found.");

      e.printStackTrace();

    }

  }

}
```

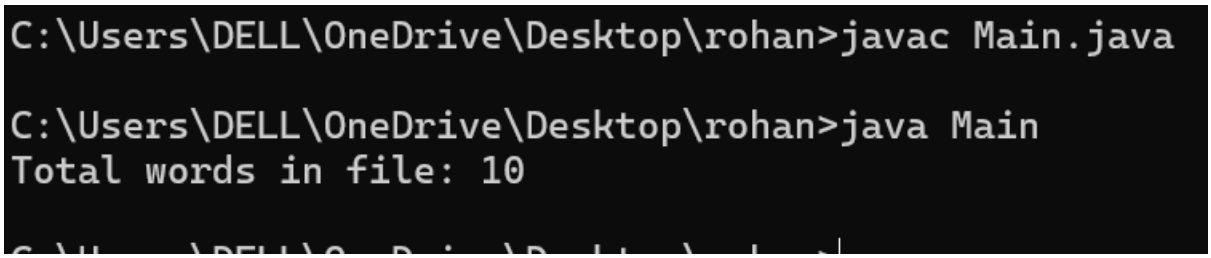
OUTPUT:



3)CODE:

```
import java.io.File;

import java.io.FileNotFoundException;

import java.util.Scanner;


public class Main {

  public static void main(String[] args) {

    try {

      File file = new File("example.txt");

      Scanner reader = new Scanner(file);

      int wordCount = 0;

      while (reader.hasNextLine()) {

        String line = reader.nextLine();

        String[] words = line.split("\\s+");

        wordCount += words.length;

      }

      reader.close();

      System.out.println("Total words in file: " + wordCount);

    } catch (FileNotFoundException e) {

      System.out.println("File not found.");

      e.printStackTrace();

    }

  }

}
```

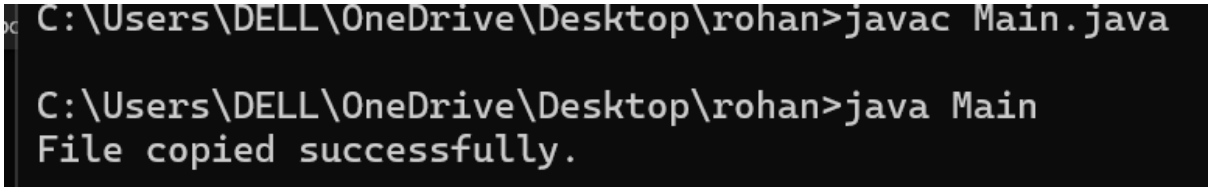
OUTPUT:



4)CODE:

```
import java.io.FileReader;

import java.io.FileWriter;

import java.io.IOException;



public class Main {

  public static void main(String[] args) {

    String sourceFile = "example.txt";

    String destinationFile = "copy_example.txt";



    try {

      FileReader reader = new FileReader(sourceFile);

      FileWriter writer = new FileWriter(destinationFile);



      int ch;

      while ((ch = reader.read()) != -1) {

        writer.write(ch);

      }



      reader.close();

      writer.close();

      System.out.println("File copied successfully.");

    } catch (IOException e) {

      System.out.println("An error occurred.");

      e.printStackTrace();

    }

  }

}
```

OUTPUT:



----------THE END----------