SDP

DAY 2

1. Conditional Statements:

a)Tryout: If-else:

class Account{

    public static void main(String[] args){

        double balance = 600;

        System.out.println("Amount to withdraw");

        double amount = 500;

        if (amount <= 0 || amount > balance){

           System.out.println("Withdrawal has failed");

        }else{

            balance-=amount;

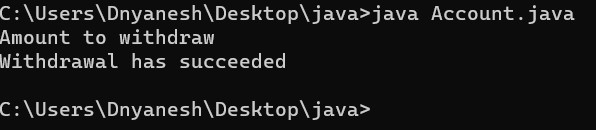
            System.out.println("Withdrawal has succeeded");

        }

    }

}

Output:



b)Tryout: if-else-if-else:

class Account{

    public static void main(String[] args){

        double balance = 600;

        System.out.println("Amount to withdraw");

        double amount = 500;

        if(amount<=0){

            System.out.println("Withdrawal has failed as the amount is negative");

        }

        else if(amount>balance){

            System.out.println("Withdrawal has failed as the balance is low");

        }

        else{

            balance-=amount;

            System.out.println("Withdrawal has succeeded");

        }

    }

}

Output:



c) Tryout: Switch case:

class Account {

  public static void main(String[] args) {

    int choice = 0;

    choice = 2;   // Input taken from user

    switch(choice) {  // choice passed to switch statement

      // choice has to match the case value to execute the statements in that case option.

      case 1:         // new record entry: newEntry()

        System.out.println("Entry deposition");

        break;

      case 2:         // displaying details of account: display()

        System.out.println("Display operation");

        break;        // break keyword is used to exit from the switch block.

      case 3:         // deposit operation: deposit()

        System.out.println("Deposit operation");

        break;

      default:        // If no case matches, default will be executed.

        System.out.println("Invalid choice");

    }

  }

}

Output:



d) Tryout: do-while loop:

import java.util.Scanner;

class Account {

  public static void main(String[] args) {

    double balance = 0;

    double minbal = 500;

    double depositAmt = 0;

    //Scanner sc = new Scanner(System.in);  uncomment when working in eclipse

    do {

      System.out.println("$100 have been added to the account");

      depositAmt +=100;                     // harcode different depositAmt values

      //depositAmt = sc.nextInt();          uncomment when working in eclipse

    } while(depositAmt < minbal);

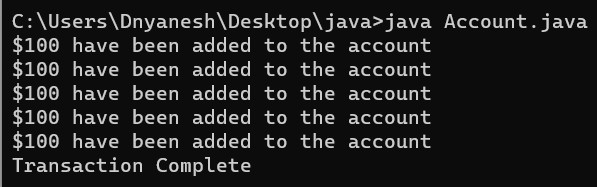
    balance = depositAmt;

    System.out.println("Transaction Complete");

  }

}

Output:



e) Tryout: While loop:

import java.util.Scanner;

class Account {

  public static void main(String[] args) {

    double balance = 0;

    double minbal = 500;

    double depositAmt = 0;

    //Scanner sc = new Scanner(System.in);      // uncomment when working in eclipse

    while(depositAmt < minbal) {

      System.out.println("$100 have been added to account");

      //depositAmt = sc.nextInt();              // harcode different depositAmt values

                                                // uncomment when working in eclipse

      depositAmt +=100;

    }

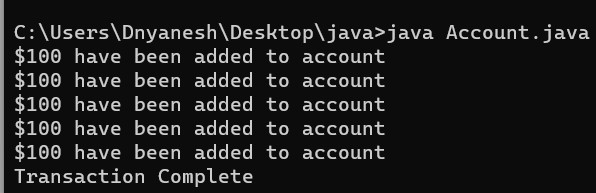
    balance = depositAmt;

    System.out.println("Transaction Complete");

  }

}

Output:



f) Tryout: For loop:

class Account {

  public static void main(String[] args) {

    double balance = 6000, rateOfInterest = 0.10, interest = 0;

    double withdrawal = 500, deposit = 600;

    for(int i = 1; i <= 12; ++i) {

      balance += deposit;

      balance -= deposit;

      interest = balance \* rateOfInterest;

      balance += interest;

      System.out.println("The interest for the month " + i + " is " + interest);

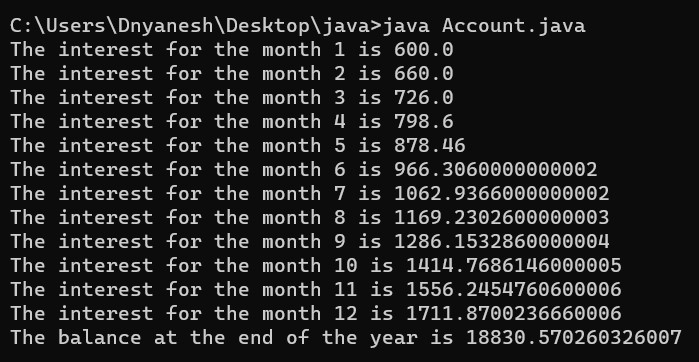
    }

    System.out.println("The balance at the end of the year is " + balance);

  }

}

Output:



h) Exercise 1:

public class ReverseCalculator {

    // Method to reverse a number using a while loop

    public static int reverseWithWhileLoop(int number) {

        int reversedNumber = 0;

        while (number != 0) {

            int digit = number % 10;

            reversedNumber = reversedNumber \* 10 + digit;

            number /= 10;

        }

        return reversedNumber;

    }

    // Method to reverse a number using a for loop

    public static int reverseWithForLoop(int number) {

        int reversedNumber = 0;

        for (; number != 0; number /= 10) {

            int digit = number % 10;

            reversedNumber = reversedNumber \* 10 + digit;

        }

        return reversedNumber;

    }

    // Method to reverse a number using a do-while loop

    public static int reverseWithDoWhileLoop(int number) {

        int reversedNumber = 0;

        do {

            int digit = number % 10;

            reversedNumber = reversedNumber \* 10 + digit;

            number /= 10;

        } while (number != 0);

        return reversedNumber;

    }

    public static void main(String[] args) {

        int number = 27;

        // Using while loop

        int reversedWithWhileLoop = reverseWithWhileLoop(number);

        System.out.println("Reverse using while loop: " + reversedWithWhileLoop);

        // Using for loop

        int reversedWithForLoop = reverseWithForLoop(number);

        System.out.println("Reverse using for loop: " + reversedWithForLoop);

        // Using do-while loop

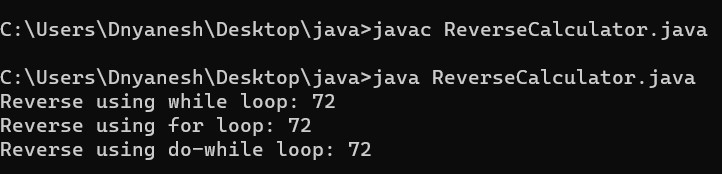
        int reversedWithDoWhileLoop = reverseWithDoWhileLoop(number);

        System.out.println("Reverse using do-while loop: " + reversedWithDoWhileLoop);

    }

}

Output:



1. Exercise 2:

public class FunCalculator {

    // Method to check if a number is an Armstrong number

    public static boolean isArmstrongNumber(int number) {

        int originalNumber = number;

        int sum = 0;

        while (number > 0) {

            int digit = number % 10;

            sum += Math.pow(digit, 3);

            number /= 10;

        }

        return sum == originalNumber;

    }

    // Method to check if a number is a lucky number

    public static boolean isLuckyNumber(int number) {

        int sum = 0;

        boolean alternateDigit = false;

        while (number > 0) {

            int digit = number % 10;

            if (alternateDigit) {

                sum += digit \* digit;

            }

            alternateDigit = !alternateDigit;

            number /= 10;

        }

        return sum % 9 == 0;

    }

    public static void main(String[] args) {

        int armstrongNum = 153;

        int luckyNum = 2864;

        // Check if the number is an Armstrong number

        if (isArmstrongNumber(armstrongNum)) {

            System.out.println(armstrongNum + " is an Armstrong number.");

        } else {

            System.out.println(armstrongNum + " is not an Armstrong number.");

        }

        // Check if the number is a lucky number

        if (isLuckyNumber(luckyNum)) {

            System.out.println(luckyNum + " is a lucky number.");

        } else {

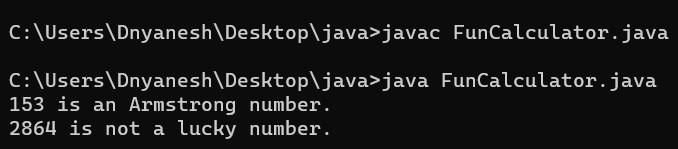
            System.out.println(luckyNum + " is not a lucky number.");

        }

    }

}

Output:



1. Arrays
2. Tryout: 1D Arrays

class Bank {

  public static void main(String[] args) {

    int[] phone = new int[3];   // primitive type array

    phone[0] = 7120686;

    phone[1] = 7120687;       // inserting element in an array

    phone[2] = 7120684;

    int count = 1;

    for(int i = 0; i < phone.length; ++i) {   // traversing an array

      System.out.println("phone number " + count + ": " + phone[i]);

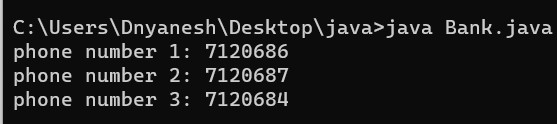
      ++count;

    }

  }

}

Output:



1. Tryout: Reference type arrays:

class bank{

    public static void main(String[] args){

        Customer[] customer = new Customer[2]; //Reference type Array

        Customer customer1 = new Customer("Anil", "Acc12345");

        Customer customer2 = new Customer("Ajay", "Acc12346");

        customer[0] = customer1; //storing in the array

        customer[1] = customer2;

        for(int i=0;i<customer.length;i++){ //traversing the array

            Customer customeObject = customer[i]; //retrieving customer Object

            String name = customeObject.displayCustomerName();

            System.out.println("the customer name is..."+name);

        }

    }

}

class Customer{

    private String name;

    private String customerId;

    Customer(String uname, String ucustomerId){

        name = uname;

        customerId = ucustomerId;

    }

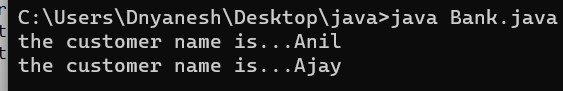
    public String displayCustomerName(){

        return name;

    }

}

Output:



1. Exercise:

public class EmployeeRecord {

    private double[] salaries;

    public EmployeeRecord(double[] salaries) {

        this.salaries = salaries;

    }

    public void displaySalaries() {

        System.out.println("Employee Salaries:");

        for (int i = 0; i < salaries.length; i++) {

            System.out.println("Employee " + (i + 1) + ": $" + salaries[i]);

        }

    }

    public double calculateAverageSalary() {

        double total = 0;

        for (double salary : salaries) {

            total += salary;

        }

        return total / salaries.length;

    }

    public int countAboveAverage() {

        double average = calculateAverageSalary();

        int count = 0;

        for (double salary : salaries) {

            if (salary > average) {

                count++;

            }

        }

        return count;

    }

    public int countBelowAverage() {

        double average = calculateAverageSalary();

        int count = 0;

        for (double salary : salaries) {

            if (salary < average) {

                count++;

            }

        }

        return count;

    }

    public static void main(String[] args) {

        double[] salary = {23500.0, 25080.0, 28760.0, 22340.0, 19890.0};

        EmployeeRecord record = new EmployeeRecord(salary);

        record.displaySalaries();

        double averageSalary = record.calculateAverageSalary();

        System.out.println("Average Salary: $" + averageSalary);

        int aboveAverageCount = record.countAboveAverage();

        System.out.println("Number of employees earning above average: " + aboveAverageCount);

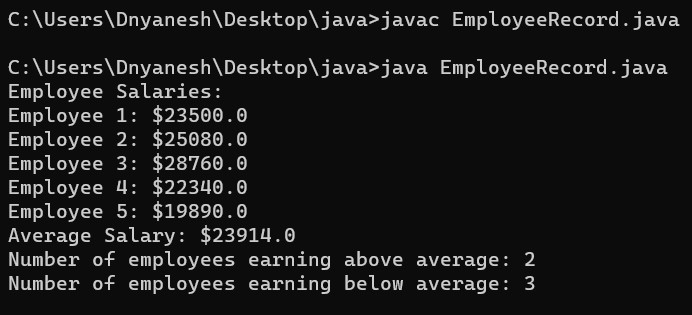
        int belowAverageCount = record.countBelowAverage();

        System.out.println("Number of employees earning below average: " + belowAverageCount);

    }

}

Output:



1. Enhanced For loop:
2. Tryout:

class Acccount{

    public static void main(String[] args){

        double balance = 6000;

        double rateOfInterest = 0.10;

        double interest = 0;

        double withdrawal = 500;

        double deposit = 600;

        int[] arr = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12};

        for(int i: arr) {   // The iteration in the loop happens automatically. The value is assigned to

                   //variable i from the array in every iteration of the loop.

            balance += deposit;   // Loop will repeat the statements in its body till the last element is reached in the array.

            balance -= withdrawal;

            interest = balance \* rateOfInterest;

            balance += interest;

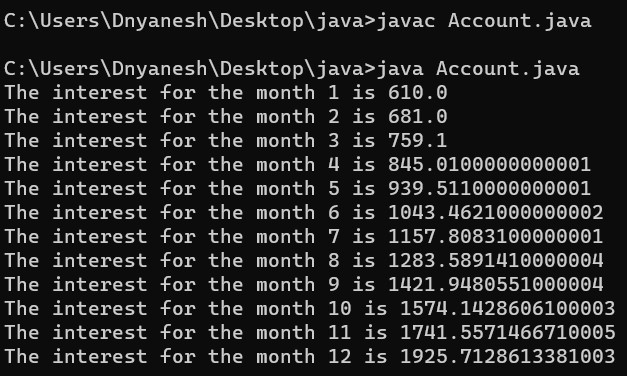
            System.out.println("The interest for the month " + i + " is " + interest);

            }

        }

}

Output:



1. Exercise:

public class SubjectMarks {

    private int[] marksArray;

    public SubjectMarks() {

        marksArray = new int[5];

    }

    public void storeMarks(int[] marks) {

        if (marks.length == 5) {

            System.arraycopy(marks, 0, marksArray, 0, 5);

            System.out.println("Marks stored successfully.");

        } else {

            System.out.println("Please provide marks for all 5 subjects.");

        }

    }

    public void displayMarks() {

        System.out.println("Subject Marks:");

        int subjectNumber = 1;

        for (int mark : marksArray) {

            System.out.println("Subject " + subjectNumber + ": " + mark);

            subjectNumber++;

        }

    }

    public static void main(String[] args) {

        SubjectMarks subjectMarks = new SubjectMarks();

        int[] marks = {75, 82, 90, 68, 88}; // Sample marks for 5 subjects

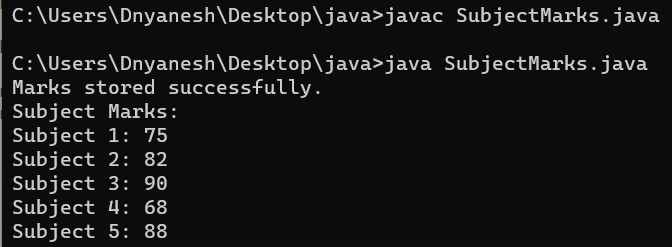
        subjectMarks.storeMarks(marks);

        subjectMarks.displayMarks();

    }

}

Output:



1. Constructors:
2. Tryout:

class Bank {

  private String bankName, area;

  private String phoneNumber;

  Bank() {    // Default constructor

    bankName = "IBank";

    area = "Gandhi Nagar";

    phoneNumber = "9876543210";

  }

  Bank(String bname, String barea, String phoneNo) {  // Parameterized constructor

    bankName = bname;

    area = barea;

    phoneNumber = phoneNo;

  }

  void displayBankDetails(){

      System.out.println("bank Name: " + bankName);

      System.out.println("Area of bank: " + area);

      System.out.println("Phone number of bank: " + phoneNumber);

  }

  public static void main(String[] args){

      Bank bank1 = new Bank();  //call default constructor

      Bank bank2 = new Bank("IBank", "Jaydev Nagar", "8876543219"); //call Parameterized constructor

      bank1.displayBankDetails();

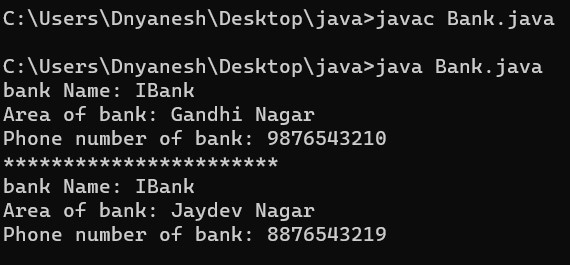
      System.out.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

      bank2.displayBankDetails();

  }

}

Output:



1. Exercise:

public class Chocolate {

    private double defaultWeight;

    private double defaultCost;

    public Chocolate(double defaultWeight, double defaultCost) {

        this.defaultWeight = defaultWeight;

        this.defaultCost = defaultCost;

    }

    public double getDefaultWeight() {

        return defaultWeight;

    }

    public void setDefaultWeight(double defaultWeight) {

        this.defaultWeight = defaultWeight;

    }

    public double getDefaultCost() {

        return defaultCost;

    }

    public void setDefaultCost(double defaultCost) {

        this.defaultCost = defaultCost;

    }

    public void displayChocolateDetails() {

        System.out.println("Chocolate Details:");

        System.out.println("Default Weight: " + defaultWeight + " grams");

        System.out.println("Default Cost: $" + defaultCost);

    }

    public static void main(String[] args) {

        // Creating a Chocolate object with default weight of 50 grams and default cost of $2.50

        Chocolate myChocolate = new Chocolate(50, 2.50);

        // Displaying the default details of the chocolate

        myChocolate.displayChocolateDetails();

        // Modifying the default weight and cost

        myChocolate.setDefaultWeight(60);

        myChocolate.setDefaultCost(3.00);

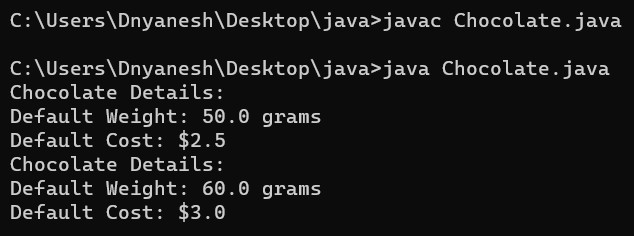
        // Displaying the updated details of the chocolate

        myChocolate.displayChocolateDetails();

    }

}

Output:



1. This Keyword:
2. Tryout:

class Bank {

    private String bankName; //instance variable

    private String area;

    private String phoneNumber;

    Bank(String bankName, String area, String phoneNumber) { // Parameterized constructor

        this.bankName = bankName;   //this keyword is used to assign

        this.area = area;           //the value for instance variables

        this.phoneNumber = phoneNumber;

    }

    void displayBankDetails(){

        System.out.println("bank Name: " + bankName);

        System.out.println("Area of bank: " + area);

        System.out.println("Phone number of bank: " + phoneNumber);

    }

    public static void main(String[] args){

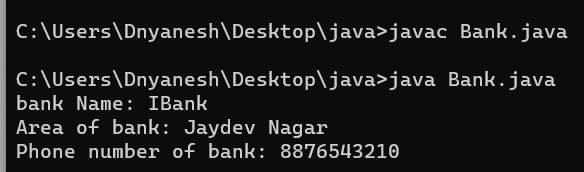
        Bank bank = new Bank("IBank", "Jaydev Nagar", "8876543210"); //call default constructor

        bank.displayBankDetails();

    }

}

Output:



1. Exercise:

public class Chocolate {

    private int barCode;

    private String name;

    private double weight;

    private double cost;

    // Parameterized constructor

    public Chocolate(int barCode, String name, double weight, double cost) {

        this.barCode = barCode;

        this.name = name;

        this.weight = weight;

        this.cost = cost;

    }

    // Default constructor

    public Chocolate() {

        this.barCode = 101;

        this.name = "Cadbury";

        this.weight = 12;

        this.cost = 10;

    }

    // Setter methods to modify values

    public void setBarCode(int barCode) {

        this.barCode = barCode;

    }

    public void setName(String name) {

        this.name = name;

    }

    public void setWeight(double weight) {

        this.weight = weight;

    }

    public void setCost(double cost) {

        this.cost = cost;

    }

    // Method to display chocolate details

    public void displayChocolateDetails() {

        System.out.println("Chocolate Details:");

        System.out.println("Barcode: " + barCode);

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

        System.out.println("Weight: " + weight + " grams");

        System.out.println("Cost: $" + cost);

    }

    public static void main(String[] args) {

        // Creating a Chocolate object using parameterized constructor

        Chocolate myChocolate = new Chocolate(101, "Cadbury", 12, 10);

        // Displaying default chocolate details

        myChocolate.displayChocolateDetails();

        // Modifying values using setter methods

        myChocolate.setBarCode(102);

        myChocolate.setName("Hershey's");

        myChocolate.setWeight(24);

        myChocolate.setCost(50);

        // Displaying updated chocolate details

        myChocolate.displayChocolateDetails();

        // Creating a Chocolate object using default constructor

        Chocolate defaultChocolate = new Chocolate();

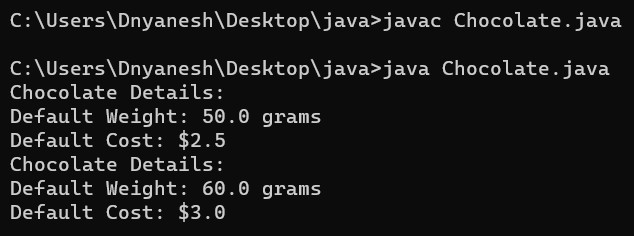
        // Displaying default chocolate details

        defaultChocolate.displayChocolateDetails();

    }

}

Output:



1. Inheritance:
2. Tryout:

class Loan {

  int tenure;

  double principal;

  float interestRate;

  String accountNumber;

  public double calculateEMI(){

      double simpleInterest = (principal\*interestRate\*tenure)/100;

      return (simpleInterest+principal)/tenure;

  }

}

class HomeLoan extends Loan {

  HomeLoan() {

    tenure = 5; //reusing super class member variables

    principal = 20000;

    interestRate = 8.5f;

    accountNumber = "Acc12345";

  }

public static void main(String[] args) {

    HomeLoan hloan = new HomeLoan();

    double amount = hloan.calculateEMI();   // sub class Object

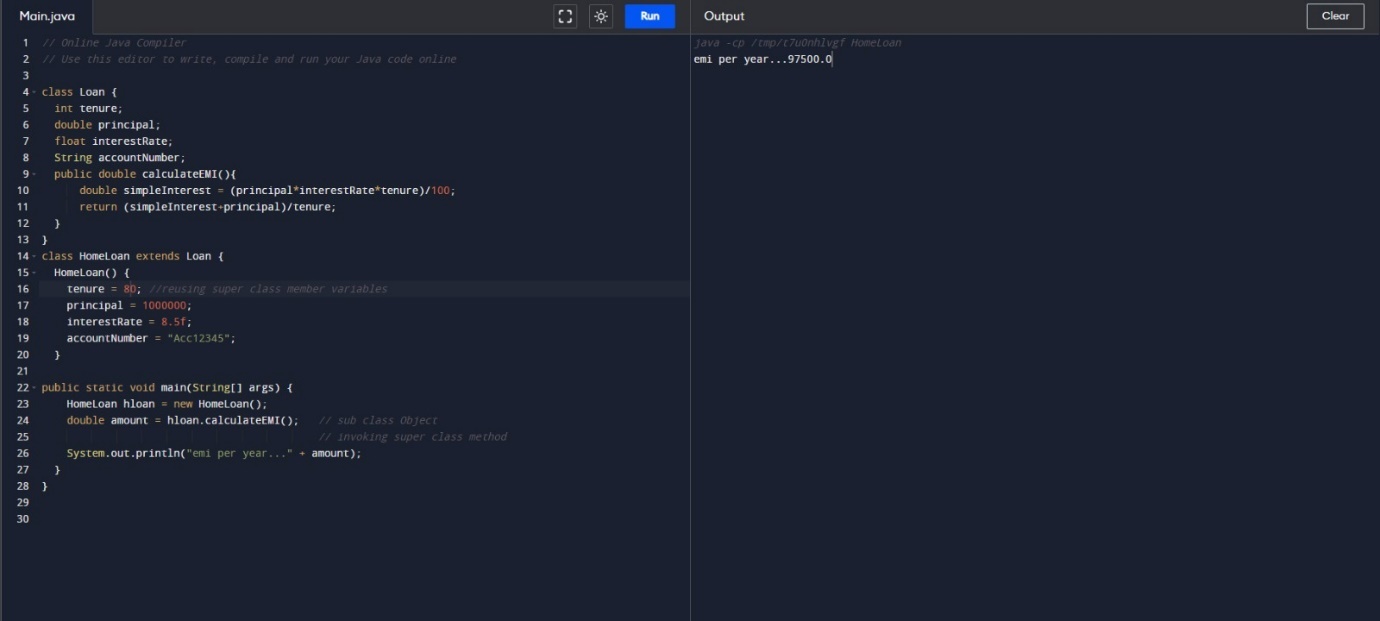
                                            // invoking super class method

    System.out.println("emi per year..." + amount);

  }

}

Output:



1. Exercise:

public class EmployeeRecords {

    public static void main(String[] args) {

        // For PermanentEmployee

        PermanentEmployee permanentEmp = new PermanentEmployee("John Doe", 101, 5000, 1500, 2000);

        permanentEmp.calculateSalary();

        // Displaying details of PermanentEmployee

        System.out.println("Details of Permanent Employee:");

        permanentEmp.displayDetails();

        System.out.println();

        // For ContractEmployee

        ContractEmployee contractEmp = new ContractEmployee("Alice Smith", 102, 40, 15);

        contractEmp.calculateSalary();

        // Displaying details of ContractEmployee

        System.out.println("Details of Contract Employee:");

        contractEmp.displayDetails();

    }

}

class Employee {

    protected String name;

    protected int employeeId;

    protected double salary;

    public Employee(String name, int employeeId) {

        this.name = name;

        this.employeeId = employeeId;

        this.salary = 0.0;

    }

    public void displayDetails() {

        System.out.println("Employee ID: " + employeeId);

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

        System.out.println("Salary: $" + salary);

    }

}

class PermanentEmployee extends Employee {

    private double basicPay;

    private double variableComponent;

    private double HRA;

    public PermanentEmployee(String name, int employeeId, double basicPay, double variableComponent, double HRA) {

        super(name, employeeId);

        this.basicPay = basicPay;

        this.variableComponent = variableComponent;

        this.HRA = HRA;

    }

    public void calculateSalary() {

        salary = variableComponent + basicPay + HRA;

    }

}

class ContractEmployee extends Employee {

    private double totalHours;

    private double wages;

    public ContractEmployee(String name, int employeeId, double totalHours, double wages) {

        super(name, employeeId);

        this.totalHours = totalHours;

        this.wages = wages;

    }

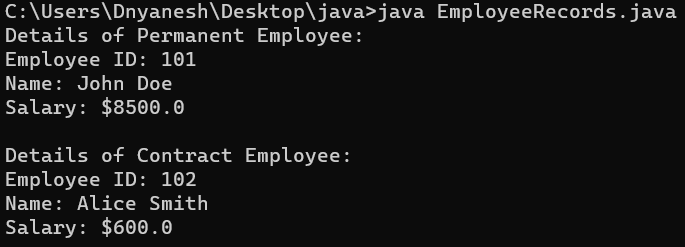
    public void calculateSalary() {

        salary = totalHours \* wages;

    }

}

Output:



1. Polymorphism:
2. Tryout1: Overloading:-

class Loan{

    private float interest;

    Loan(){

        interest = 8.5f;

    }

    //calculateEMI overloaded methods

    public double calculateEMI(int tenure, double principal){

        double simpleInterest = (principal \* interest \* tenure) / 100;

        return (simpleInterest+principal)/tenure;

    }

    public double calculateEMI(double principal, int tenure){

        double simpleInterest = (principal \* interest \* tenure) / 100;

        return (simpleInterest+principal)/tenure;

    }

    public double calculateEMI(int tenure, double principal, float interest){

        double simpleInterest = (principal \* interest \* tenure) / 100;

        return (simpleInterest+principal)/tenure;

    }

    public static void main(String[] args){

        Loan loan = new Loan();

        double result = loan.calculateEMI(20000d, 5); //d means double

        double value = loan.calculateEMI(5, 20000d);

        double val = loan.calculateEMI(5, 20000, 9.5f); // f means float

        System.out.println("EMI per year is..." + result);

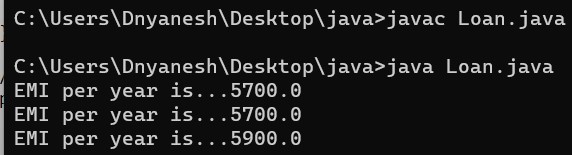
        System.out.println("EMI per year is..." + value);

        System.out.println("EMI per year is..." + val);

    }

}

Output:



1. Tryout2: Overriding:-

class ExecuteLoan{

    public static void main(String[] args){

        Loan loan = null;

        loan = new HomeLoan();      //Runtime Polymorphism

        double hloan = loan.calculateEMI(2000000);

        loan = new VehicleLoan();   //sup class reference holding sub class Object

        double vloan = loan.calculateEMI(100000);

        System.out.println("Home loan emi per year is..." + hloan);

        System.out.println("Vehicle loan emi per year is..." + vloan);

    }

}

class Loan{

    public double calculateEMI(double principal){

        double simpleInterest = (principal \* 8.5 \* 5) / 100;

        return (simpleInterest+principal)/5;

    }

}

class HomeLoan extends Loan {

    // method overriden

    public double calculateEMI(double principal){

        int additionalTax = 200;

        double simpleInterest = (principal \* 7.5 \* 20) / 100;

        double emi = (simpleInterest + principal) / 20;

        return emi+additionalTax;

    }

}

class VehicleLoan extends Loan {

    // method overriden

    public double calculateEMI(double principal){

        int additionalTax = 200;

        double simpleInterest = (principal \* 9.5 \* 10) / 100;

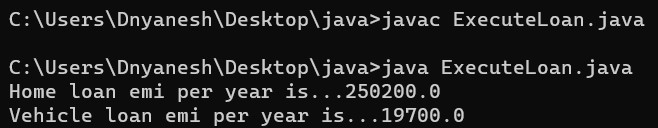
        double emi = (simpleInterest + principal) / 10;

        return emi+additionalTax;

    }

}

Output:



1. Exercise 1:

public class RegistrationTester {

    public static void main(String[] args) {

        // First case

        Registration user1 = new Registration("Kevin", "9452425421", "7676765252");

        user1.setPassportDetails("MN989IN");

        user1.displayRegistrationDetails();

        // Second case

        Registration user2 = new Registration("Julias", "2345615451", "6763562562");

        user2.setLicenseAndPanDetails("123", "PN7878");

        user2.displayRegistrationDetails();

        // Third case

        Registration user3 = new Registration("Jammy", "9634524353", "9887373737");

        user3.setVoterIdAndLicenseDetails("45453", "765");

        user3.displayRegistrationDetails();

    }

}

class Registration {

    private String customerName;

    private String passportNumber;

    private String licenseNumber;

    private String panCardNumber;

    private String voterId;

    private String telephoneNumber1;

    private String telephoneNumber2;

    // Constructor with mandatory fields

    public Registration(String customerName, String telephoneNumber1, String telephoneNumber2) {

        this.customerName = customerName;

        this.telephoneNumber1 = telephoneNumber1;

        this.telephoneNumber2 = telephoneNumber2;

    }

    // Method to set passport details

    public void setPassportDetails(String passportNumber) {

        this.passportNumber = passportNumber;

    }

    // Method to set license and pan card details

    public void setLicenseAndPanDetails(String licenseNumber, String panCardNumber) {

        this.licenseNumber = licenseNumber;

        this.panCardNumber = panCardNumber;

    }

    // Method to set voter ID and license details

    public void setVoterIdAndLicenseDetails(String voterId, String licenseNumber) {

        this.voterId = voterId;

        this.licenseNumber = licenseNumber;

    }

    // Method to set pan card and voter ID details

    public void setPanCardAndVoterIdDetails(String panCardNumber, String voterId) {

        this.panCardNumber = panCardNumber;

        this.voterId = voterId;

    }

    // Method to display registration details

    public void displayRegistrationDetails() {

        System.out.println("Congratulations " + customerName + "!!! you have been successfully registered for our services with the following details:");

        if (passportNumber != null) {

            System.out.println("Passport number: " + passportNumber);

        } else if (licenseNumber != null && panCardNumber != null) {

            System.out.println("License number: " + licenseNumber);

            System.out.println("Pan card number: " + panCardNumber);

        } else if (voterId != null && licenseNumber != null) {

            System.out.println("Voter ID: " + voterId);

            System.out.println("License number: " + licenseNumber);

        } else if (panCardNumber != null && voterId != null) {

            System.out.println("Pan card number: " + panCardNumber);

            System.out.println("Voter ID: " + voterId);

        }

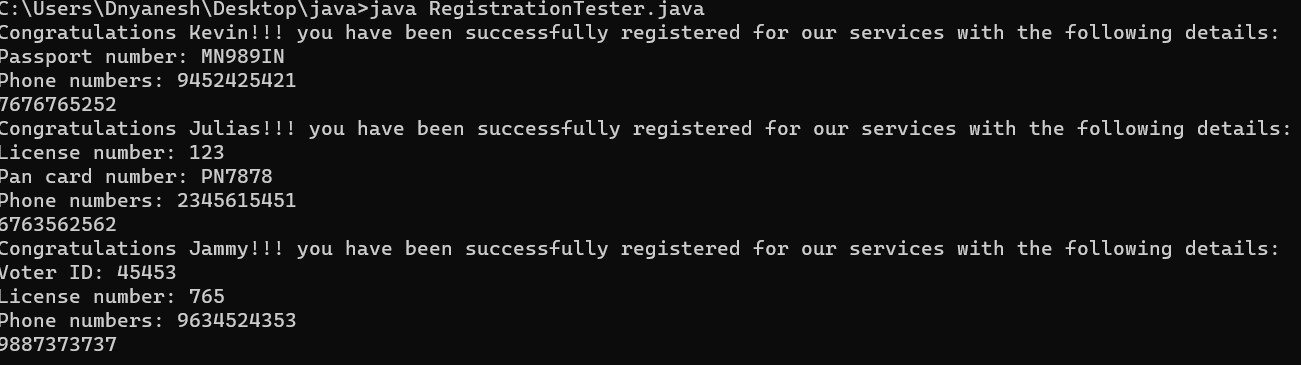
        System.out.println("Phone numbers: " + telephoneNumber1);

        System.out.println(telephoneNumber2);

    }

}

Output:



1. Exercise 2:

public class RegistrationTester {

    public static void main(String[] args) {

        // Input for User 1

        Registration user1 = new Registration("Kevin", "9452425421", "7676765252");

        user1.setPassportNumber("MN989IN");

        user1.displayRegistrationDetails();

        // Input for User 2

        Registration user2 = new Registration("Julias", "2345615451", "6763562562");

        user2.setLicenseAndPanDetails("123", "PN7878");

        user2.displayRegistrationDetails();

        // Input for User 3

        Registration user3 = new Registration("Jammy", "9634524353", "9887373737");

        user3.setVoterIdAndLicenseDetails("45453", "765");

        user3.displayRegistrationDetails();

        // Input for User 4

        Registration user4 = new Registration("Rose", "9867456367", "7645367356");

        user4.setPanCardAndVoterIdDetails("PN8934", "34356");

        user4.displayRegistrationDetails();

    }

}

class Registration {

    private String customerName;

    private String passportNumber;

    private String licenseNumber;

    private String panCardNumber;

    private String voterId;

    private String telephoneNumber1;

    private String telephoneNumber2;

    // Constructor to initialize mandatory fields

    public Registration(String customerName, String telephoneNumber1, String telephoneNumber2) {

        this.customerName = customerName;

        this.telephoneNumber1 = telephoneNumber1;

        this.telephoneNumber2 = telephoneNumber2;

    }

    // Setter methods for different ID types

    public void setPassportNumber(String passportNumber) {

        this.passportNumber = passportNumber;

    }

    public void setLicenseAndPanDetails(String licenseNumber, String panCardNumber) {

        this.licenseNumber = licenseNumber;

        this.panCardNumber = panCardNumber;

    }

    public void setVoterIdAndLicenseDetails(String voterId, String licenseNumber) {

        this.voterId = voterId;

        this.licenseNumber = licenseNumber;

    }

    public void setPanCardAndVoterIdDetails(String panCardNumber, String voterId) {

        this.panCardNumber = panCardNumber;

        this.voterId = voterId;

    }

    // Method to display registration details

    public void displayRegistrationDetails() {

        System.out.println("Congratulations " + customerName + "!!! you have been successfully registered for our services with the following details:");

        if (passportNumber != null) {

            System.out.println("Passport number: " + passportNumber);

        } else if (licenseNumber != null && panCardNumber != null) {

            System.out.println("License number: " + licenseNumber);

            System.out.println("Pan card number: " + panCardNumber);

        } else if (voterId != null && licenseNumber != null) {

            System.out.println("Voter id: " + voterId);

            System.out.println("License number: " + licenseNumber);

        } else if (panCardNumber != null && voterId != null) {

            System.out.println("Pan card number: " + panCardNumber);

            System.out.println("Voter id: " + voterId);

        }

        System.out.println("Phone numbers: " + telephoneNumber1);

        System.out.println(telephoneNumber2);

    }

}

Output:

