**1st**

**class and object**

```
import java.util.Scanner;

public class Complex {
   private double real;          // instance variable
   private double img;

   public Complex(){ //Default constructor
      this.real=0;
      this.img=0;
   }

   public Complex (double real, double img){ //Parameterized Constructor
      this.real = real;
      this.img = img;
   }
   //Static Methods
   public static Complex add(Complex n1,Complex n2){ //Addition
      Complex temp = new Complex();
      temp.real = n1.real + n2.real;
      temp.img = n1.img + n2.img;
      return (temp);
   }

   public static Complex sub(Complex n1,Complex n2){ //Subtraction
      Complex temp = new Complex();
      temp.real = n1.real - n2.real;
      temp.img = n1.img - n2.img;
```

```
    return (temp);
  }

  public static Complex mult(Complex n1,Complex n2){ //Multiplication
    Complex temp=new Complex();
    temp.real=(n1.real*n2.real)+(n1.img*n2.img*(-1));
    temp.img=(n1.img*n2.real)+(n2.img*n1.real);
    return (temp);
  }

  public static void display(Complex n){ //Display
    if(n.img>=0){
      System.out.printf("\t = %.1f + %.1fi",n.real,n.img);
    }
    else
      System.out.printf("\t = %.1f  %.1fi",n.real,n.img);

    }

  public static void main(String []args){
    Scanner sc=new Scanner(System.in);//This allows user input for real and imaginary parts of two complex numbers
    System.out.println("ComplexNo. 1: ");//Input for First Complex Number
    System.out.print("\nEnter real: ");
    double r1=sc.nextDouble();
    System.out.print("\nEnter img: ");
    double i1=sc.nextDouble();

    System.out.println("\n\nComplexNo. 2: ");//Input for Second Complex Number
    System.out.print("\nEnter real: ");
    double r2=sc.nextDouble();
```

```
        System.out.print("\nEnter img: ");
        double i2=sc.nextDouble();
        Complex n1 = new Complex(r1,i1),n2 = new Complex(r2,i2),temp;//Create Two Complex Objects
        //Perform and Display Operations
        temp = add(n1,n2);//Addition
        System.out.println("\nAddition is : ");
        display(temp);

        temp = sub(n1,n2);//Subtraction
        System.out.println("\nSubtraction is : ");
        display(temp);

        temp = mult(n1,n2);//Multiplication
        System.out.println("\nMultiplication is : ");
        display(temp);
    }
}
```

2 nd ; polymorphism

```
package polymorphism;
import java.util.Scanner;
//Base class: Publication
class publication1 {
String title;
double price;
int copies;
// Constructor
public publication1(String title, double price, int copies) {
  this.title = title;
  this.price = price;
```

```
  this.copies = copies;
}
// Method to sell copies
public void saleCopy(int numberOfCopies) {
  if (numberOfCopies <= copies) {
     copies = copies- numberOfCopies;
     System.out.println(numberOfCopies + " copies of " + title + " sold.");
  } else {
     System.out.println("Not enough copies available!");
  }
}

// Method to calculate total sale
public double totalSale(int numberOfCopies) {
  return price * numberOfCopies;
}
}

//Book class
class Book extends publication1 {
String author;

// Constructor
public Book(String title, String author, double price, int copies) {
  super(title, price, copies);
  this.author = author;
}

// Method to order more copies
public void orderCopies(int newCopies) {
  copies += newCopies;
```

```
  System.out.println(newCopies + " more copies of " + title + " by " + author + " ordered.");
}
}

//Magazine class
class Magazine extends publication1 {
String currentIssue;

// Constructor
public Magazine(String title, double price, int copies, String currentIssue) {
  super(title, price, copies);
  this.currentIssue = currentIssue;
}

// Method to order more copies
public void orderQty(int newCopies) {
  copies += newCopies;
  System.out.println(newCopies + " more copies of " + title + " (Issue: " + currentIssue + ") ordered.");
}

// Method to receive new issue
public void receiveIssue(String newIssue) {
  currentIssue = newIssue;
  System.out.println("New issue " + newIssue + " of " + title + " received.");
}
}


//Main class to handle user input and operations

public class publication{
```

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

  // Input and create a book
  System.out.println("Enter details for the book:");
  System.out.print("Title: ");
  String bookTitle = scanner.nextLine();
  System.out.print("Author: ");
  String author = scanner.nextLine();
  System.out.print("Price: ");
  double bookPrice = scanner.nextDouble();
  System.out.print("Number of copies: ");
  int bookCopies = scanner.nextInt();
  Book book = new Book(bookTitle, author, bookPrice, bookCopies);

  // Input and create a magazine
  scanner.nextLine(); // Consume the remaining newline
  System.out.println("\nEnter details for the magazine:");
  System.out.print("Title: ");
  String magazineTitle = scanner.nextLine();
  System.out.print("Price: ");
  double magazinePrice = scanner.nextDouble();
  System.out.print("Number of copies: ");
  int magazineCopies = scanner.nextInt();
  scanner.nextLine(); // Consume newline
  System.out.print("Current Issue: ");
  String currentIssue = scanner.nextLine();
  Magazine magazine = new Magazine(magazineTitle, magazinePrice, magazineCopies, currentIssue);

  // Perform operations
  System.out.print("\nHow many copies of the book would you like to order? ");
```

```
  int bookOrderCopies = scanner.nextInt();
  book.orderCopies(bookOrderCopies);
  book.saleCopy(bookOrderCopies);
  double totalBookSale = book.totalSale(bookOrderCopies);

  System.out.print("\nHow many copies of the magazine would you like to order? ");
  int magazineOrderCopies = scanner.nextInt();
  magazine.orderQty(magazineOrderCopies);
  magazine.saleCopy(magazineOrderCopies);
  double totalMagazineSale = magazine.totalSale(magazineOrderCopies);

  // Display total sales
  double totalSale = totalBookSale + totalMagazineSale;
  System.out.println("\nTotal sale of publications: $" + totalSale);

  scanner.close();
}
}
```

```
//3 inheritance

import java.util.Scanner;
public class Employee{
   String Emp_name,Emp_id,Emp_Address;
   String Mail_id,Mob_no;
   public void input(){
      Scanner sc = new Scanner(System.in);
```

```
        System.out.print("Enter Employee ID::");
        Emp_id=sc.nextLine();
        System.out.print("Enter Employee Name::");
        Emp_name=sc.nextLine();
        System.out.print("Enter Employee Address::");
        Emp_Address=sc.nextLine();
        System.out.print("Enter Employee Mail ID::");
        Mail_id=sc.nextLine();
        System.out.print("Enter Employee Mob. No.::");
        Mob_no=sc.next();
    }
    public void display(){
        System.out.println("Employee Id: "+Emp_id+"\nEmployee name: "+Emp_name+"\nEmployee
Address: "+Emp_Address+"\nMail Id: "+Mail_id+"\nEmployee Mob. no.: "+Mob_no);
    }
public static void main(String[] args) {
        double Base_Pay;
        int choice;
        Scanner obj=new Scanner(System.in);
        do{
            System.out.println("\n**Employee Management System");
            System.out.println("1. Programmer\n2. Team Lead\n3. Assistant Project Manager\n4. Project
Manager\n5. Exit ");


            System.out.println("Enter your choice: ");
            choice = obj.nextInt();
            switch (choice) {
                case 1:
                    Programmer p1=new Programmer();
                    System.out.println("Enter Information for Employee: ");
                    p1.input();
                    System.out.println("Basic Pay of Programmer: ");
```

```
    Base_Pay=obj.nextDouble();
    p1.set_basicPay(Base_Pay);
    p1.calculate_salary();
    p1.display();
    break;
case 2:
    Team_Lead p2=new Team_Lead();
    System.out.println("Enter Information for Employee: ");
    p2.input();
    System.out.println("Basic Pay of Team Lead: ");
    Base_Pay=obj.nextDouble();
    p2.set_basicPay(Base_Pay);
    p2.calculate_salary();
    p2.dispaly();
    break;

case 3:
    Assistant_Project_Manager p3=new Assistant_Project_Manager();
    System.out.println("Enter Information for Employee: ");
    p3.input();
    System.out.println("Basic Pay of Assistant Project Manager: ");
    Base_Pay=obj.nextDouble();
    p3.set_basicPay(Base_Pay);
    p3.calculate_salary();
    p3.dispaly();
    break;

case 4:
    Project_Manager p4=new Project_Manager();
    System.out.println("Enter Information for Employee: ");
    p4.input();
```

```
                System.out.println("Basic Pay of Project Manager: ");
                Base_Pay=obj.nextDouble();
                p4.set_basicPay(Base_Pay);
                p4.calculate_salary();
                p4.dispaly();
                break;
            default:
                break;
          }
       }while(choice!=5);
       obj.close();
    }

}



class Programmer extends Employee{
    double Basic_Pay;
    double gross_sal,net_sal;
    public Programmer(){
       super();
       Basic_Pay=0;
    }
    public void set_basicPay(double bp){
       Basic_Pay=bp;
    }
    public void calculate_salary(){
       /* 97% DA
        * 10% HRA
        * 12% PF
```

```
         * 0.1% staff club fund
         */
        // Gross salary = Basic Pay + 97% Basic Pay
        gross_sal=Basic_Pay+(Basic_Pay*0.97);

        // Net salary = Gross salary -(10% Basic Pay + 12% Basic Pay + 0.1% Basic Pay)
        net_sal=gross_sal-(0.1*Basic_Pay+0.12*Basic_Pay+0.001*Basic_Pay);
    }
    @Override
    public void display(){
        System.out.println("Baisc Pay::"+Basic_Pay);
        System.out.println("DA::"+Basic_Pay*0.97);
        System.out.println("Gross Salary::"+gross_sal);
        System.out.println("PF::"+Basic_Pay*0.12);
        System.out.println("Baisc Pay::"+Basic_Pay*0.1);
        System.out.println("Staff Club Fund::"+Basic_Pay*0.001);
        System.out.println("Net Salary::"+net_sal);
    }
}

class Team_Lead extends Employee{
    double Basic_Pay;
    double gross_sal;
    double net_sal;
    public Team_Lead(){
        super();
        Basic_Pay=0;
    }
    public void set_basicPay(double bp){
        Basic_Pay=bp;
    }
```

```
    public void calculate_salary(){
        gross_sal=Basic_Pay+0.97*Basic_Pay;
        net_sal=gross_sal-(0.1*Basic_Pay+0.12*Basic_Pay+0.001*Basic_Pay);
    }
    public void dispaly(){
        System.out.println("Baisc Pay::"+Basic_Pay);
        System.out.println("DA::"+Basic_Pay*0.97);
        System.out.println("Gross Salary::"+gross_sal);
        System.out.println("PF::"+Basic_Pay*0.12);
        System.out.println("Baisc Pay::"+Basic_Pay*0.1);
        System.out.println("Staff Club Fund::"+Basic_Pay*0.001);
        System.out.println("Net Salary::"+net_sal);

    }
}

class Assistant_Project_Manager extends Employee{
    double Basic_Pay;
    double gross_sal;
    double net_sal;
    public Assistant_Project_Manager(){
        super();
        Basic_Pay=0;
    }
    public void set_basicPay(double bp){
        Basic_Pay=bp;
    }
    public void calculate_salary(){
        gross_sal=Basic_Pay+0.97*Basic_Pay;
        net_sal=gross_sal-(0.1*Basic_Pay+0.12*Basic_Pay+0.001*Basic_Pay);
    }
```

```
    public void dispaly(){
        System.out.println("Baisc Pay::"+Basic_Pay);
        System.out.println("DA::"+Basic_Pay*0.97);
        System.out.println("Gross Salary::"+gross_sal);
        System.out.println("PF::"+Basic_Pay*0.12);
        System.out.println("Baisc Pay::"+Basic_Pay*0.1);
        System.out.println("Staff Club Fund::"+Basic_Pay*0.001);
        System.out.println("Net Salary::"+net_sal);

    }
}

class Project_Manager extends Employee{
    double Basic_Pay;
    double gross_sal;
    double net_sal;
    public Project_Manager(){
        Basic_Pay=0;
    }
    public void set_basicPay(double bp){
        Basic_Pay=bp;
    }
    public void calculate_salary(){
        gross_sal=Basic_Pay+0.97*Basic_Pay;
        net_sal=gross_sal-(0.1*Basic_Pay+0.12*Basic_Pay+0.001*Basic_Pay);
    }
    public void dispaly(){
        System.out.println("Baisc Pay::"+Basic_Pay);
        System.out.println("DA::"+Basic_Pay*0.97);
        System.out.println("Gross Salary::"+gross_sal);
        System.out.println("PF::"+Basic_Pay*0.12);
```

```
        System.out.println("Baisc Pay::"+Basic_Pay*0.1);
        System.out.println("Staff Club Fund::"+Basic_Pay*0.001);
        System.out.println("Net Salary::"+net_sal);

    }
}
```

4: dynamic binding

```
//4 dynamic binding

import java.util.Scanner;
abstract class Shape {
    double length,height;
    abstract double compute_area();

    public static void main(String[]args){
        double base,height,len,wid,area;
        Triangle t1=new Triangle();
        Rectangle r1=new Rectangle();

        System.out.println("\nEnter Base of Triangle:\t");
        Scanner obj=new Scanner(System.in);
        base=obj.nextDouble();

        System.out.println("\nEnter Height of Triangle:\t");
        height=obj.nextDouble();

        t1.set_dimention(base,height);
```

```
        area=t1.compute_area();
        System.out.println("\nArea of Triangle:\t"+area);

        System.out.println("\nEnter Length of Rectangle:\t");
        len=obj.nextDouble();

        System.out.println("\nEnter Width of Triangle:\t");
        wid=obj.nextDouble();

        obj.close();

        r1.set_dimention(len,wid);
        area=r1.compute_area();
        System.out.println("\nArea of Rectangle:\t"+area);



    }
}

class Triangle extends Shape{
    void set_dimention(double b,double h){
        length=b;
        height=h;
    }
    double compute_area(){
        double area;
        area=0.5*length*height;
        return area;
    }
}
```

```
class Rectangle extends Shape{
   void set_dimention(double b,double h){
      length=b;
      height=h;
   }
   double compute_area(){
      double area;
      area=length*height;
      return area;
   }
}
```

5: interface

6: exception handling

```
import java.util.Scanner;

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

      try {
         // Take input from user
         System.out.print("Enter first number (Num1): ");
```

```
            String input1 = sc.nextLine();
            System.out.print("Enter second number (Num2): ");
            String input2 = sc.nextLine();

            // Attempt to parse integers
            int num1 = Integer.parseInt(input1);
            int num2 = Integer.parseInt(input2);

            // Division
            int result = num1 / num2;
            System.out.println("Result of division: " + result);

            // Just to demonstrate ArrayIndexOutOfBounds
            int[] arr = new int[2];
            System.out.println("Accessing third element (index 2): " + arr[2]); // will throw exception

        } catch (NumberFormatException e) {
            System.out.println("NumberFormatException: Please enter valid integers.");
        } catch (ArithmeticException e) {
            System.out.println("ArithmeticException: Cannot divide by zero.");
        } catch (ArrayIndexOutOfBoundsException e) {
            System.out.println("ArrayIndexOutOfBoundsException: Invalid array index access.");
        } finally {
            sc.close();
            System.out.println("Program execution completed.");
        }
    }
}
```

Name

More sorting options

SE- OOPL Syllabus.pdf

More actions (Alt+A)

SE OOPL Viva Questions For Lab Assignment-7 To 11.pdf

More actions (Alt+A)

ASSIGNMENT_11.java

More actions (Alt+A)

ASSIGNMENT_10.java

More actions (Alt+A)

ASSIGNMENT_9.java

More actions (Alt+A)

ASSIGNMENT_8.java

More actions (Alt+A)

ASSIGNMENT_7.java

More actions (Alt+A)

```
/*
Assignment No.7
Title:- Template

Problem Statement:- Implement a generic program using any collection class to count the number of elements in a collection that have a specific property such as even numbers, odd number, prime number and palindromes.
*/


//CODE
```

```
// FILE NAME: Assignment_7.java

import java.util.*;
import java.lang.*;
import java.io.*;
class Assignment_7
{
   static int count=0;
   static void even_odd(int num)
   {
      if (num%2==0)
      {
         System.out.println(num+" is even number");
         count+=1;
      } else
      {
         System.out.println(num+" is odd number");
      }
   }
   static void prime_no(int num)
   {
      boolean flag = false;
      for(int i = 2; i <= num/2; ++i)
      {
         if(num % i == 0)
         {
            flag = true;
            break;
         }
      }
```

```
    if (!flag) {
        System.out.println(num + " is a prime number.");
        count += 1;
    }
    else
        System.out.println(num + " is not a prime number.");

}
static void Palindrome(String str1)
{
    StringBuilder s1=new StringBuilder(str1);
    if(str1.equals(s1.reverse().toString()))
    {
        System.out.println(str1+" is palindrome");
        count+=1;

    }
    else
    {
        System.out.println(str1+" is not palindrome");
    }
}

static void check(int ch,int num)
{
    switch(ch)
    {
        case 1:
            even_odd(num);
            break;
```

```
        case 2:
            prime_no(num);
            break;

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

    }
}
static void integer_op()
{

    int element,n,ch;
    Scanner in=new Scanner(System.in);
    ArrayList<Integer>nums=new ArrayList<Integer>();
    System.out.println("Enter the number of elements:");
    n=in.nextInt();
    for(int i=0;i<n;i++)
    {
        System.out.println("Enter number:");
        element=in.nextInt();
        nums.add(element);
    }

    System.out.println("1.Even or Odd");
    System.out.println("2.Prime or not");
    System.out.println("Choose:");

    ch=in.nextInt();
    Iterator itr = nums.iterator();
    count=0;
```

```
        while(itr.hasNext())
        {
           check(ch, (int)itr.next());
        }

        switch (ch) {
           case 1 -> {
              System.out.println("The total number of even and odd numbers occured are:");
              System.out.println("The number of even number are:" + count);
              System.out.println("The number of odd number are:" + (nums.size() - count));
           }
           case 2 -> {
              System.out.println("The total number of Prime and Non-Prime numbers occured are:");
              System.out.println("The number of PRIME number are:" + count);
              System.out.println("The numbers which are NOT Prime number are:" + (nums.size() -
count));
           }
        }

    }

    static void string_op()
    {
        int m;
        String str1;
        ArrayList<String>words=new ArrayList<String>();
        Scanner in=new Scanner(System.in);
        System.out.println("Enter the number of Strings:");
        m=in.nextInt();
        for(int i=0;i<m;i++)
        {
```

```
            System.out.println("Enter string:");
            str1=in.next();
            words.add(str1);
        }
        count=0;
        for(String w:words)
        {
            Palindrome(w);
        }
        System.out.println("The number of Palindrome string are:"+count);
        System.out.println("The number of non-palindrome string is:"+(words.size()-count));

    }
    public static void main(String []args)
    {
        Scanner in=new Scanner(System.in);
        System.out.println("Choose type");
        System.out.println("1. String");
        System.out.println("2. Integer");
        int ch=in.nextInt();

        switch (ch) {
            case 1 -> string_op();
            case 2 -> integer_op();
        }


    }
}

//OUTPUT
```

```
/*
Choose type
1. String
2. Integer
1
Enter the number of Strings:
3
Enter string:
yogesh
Enter string:
gaurav
Enter string:
madam
yogesh is not palindrome
gaurav is not palindrome
madam is palindrome
The number of Palindrome string are:1
The number of non-palindrome string is:2

Choose type
1. String
2. Integer
2
Enter the number of elements:
4
Enter number:
2
Enter number:
25
Enter number:
```

50

Enter number:

75

1.Even or Odd

2.Prime or not

Choose:

1

2 is even number

25 is odd number

50 is even number

75 is odd number

The total number of even and odd numbers occured are:

The number of even number are:2

The number of odd number are:2

*/



Name

More sorting options

SE- OOPL Syllabus.pdf

More actions (Alt+A)

SE OOPL Viva Questions For Lab Assignment-7 To 11.pdf

More actions (Alt+A)

ASSIGNMENT_11.java

More actions (Alt+A)

ASSIGNMENT_10.java

More actions (Alt+A)

ASSIGNMENT_9.java

More actions (Alt+A)

ASSIGNMENT_8.java

More actions (Alt+A)

ASSIGNMENT_7.java

More actions (Alt+A)

```
/*
Assignment No.8
Title:- File Handling.

Problem Statement:- Implement a program for maintaining a database of student records using Files.
Student has Student_id, name, Roll_no, Class, marks and address. Display the data for few students.
1. Create Database
2. Display Database
3. Delete Records
4. Update Record
5. Search Record

*/

//CODE

// FILE NAME: Assignment_8.java

import java.util.*;
import java.io.*;
```

```
public class Assignment_8
{
  public static void main(String [] args)
  {
     int op;
     Scanner sc = new Scanner(System.in);
     StudentRecords sr = new StudentRecords();

     while(true)
     {
        System.out.println("1 - Add Record\n2 - Update Record\n3 - Display Record\n4 - Search Record\n5 - Delete Record\n6 - Clear All Records\n7 - Exit");
        System.out.println();
        op = sc.nextInt();
        System.out.println();
        switch(op)
        {
           case 1: sr.addRecord();
           break;

           case 2: sr.updateRecord();
           break;

           case 3: sr.displayRecords();
           break;

           case 4: sr.searchRecords();
           break;

           case 5: sr.deleteRecord();
           break;
```

```
            case 6: sr.clearRecords();
            break;

            case 7: sr.exit();
            break;

            default: System.out.println("Invalid input.");
          }

      }
   }
}


// FILE NAME: StudentRecords.java
import java.util.*;
import java.io.*;

public class StudentRecords
{
   static void addRecord()
   {
      try
      {
         BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
         PrintWriter pw = new PrintWriter(new BufferedWriter(new FileWriter("st.txt")));
         String stdName, address;

         int stdId, rollNo, std;
         float marks;
```

```
String s;
boolean addMore = false;

do
{
   System.out.print("\nEnter student's name: ");
   stdName = br.readLine();

   System.out.print("\nEnter student's id: ");
   stdId = Integer.parseInt(br.readLine());

   System.out.print("\nEnter student's roll number: ");
   rollNo = Integer.parseInt(br.readLine());

   System.out.print("\nEnter address: ");
   address = br.readLine();

   System.out.print("\nEnter student's class: ");
   std = Integer.parseInt(br.readLine());

   System.out.print("\nEnter marks: ");
   marks = Float.parseFloat(br.readLine());

   pw.println(stdName + " " + stdId + " " + rollNo + " " + address + " " + std + " " + marks);
   System.out.print("\nRecord added successfully.\nDo you want to add more?(Y/N): ");

   s = br.readLine();
   if(s.equalsIgnoreCase("Y"))
   {
      addMore = true;
   }
```

```
            else
            {
               addMore = false;
            }

         }
         while(addMore);

         pw.close();
      }
      catch(Exception e)
      {
         System.out.println(e);
      }
   }

   static void deleteRecord()
   {
      try
      {
         BufferedReader file1 = new BufferedReader(new FileReader("st.txt"));
         PrintWriter pw = new PrintWriter(new BufferedWriter(new FileWriter("st1.txt", true)));
         String name;
         int flag = 0;
         Scanner sc = new Scanner(System.in);
         System.out.print("Enter the name of the student you want to delete: ");
         String searchName = sc.nextLine();

         while((name = file1.readLine())!=null)
         {
            String[]line = name.split(" ");
```

```
    if(!searchName.equalsIgnoreCase(line[0]))
    {
       pw.println(name);
       flag = 0;
    }
    else
    {
       line[0] = "-";
       line[1] = "-";
       line[2] = "-";
       line[3] = "-";
       line[4] = "-";
       pw.println(line[0] + " "+ line[1] + " " + line[2] + " " + line[3] + " " +line[4]);
       flag = 1;
       while((name = file1.readLine())!=null)
       {
          pw.println(name);
       }
       break;
    }
}

file1.close();
pw.close();
if(flag == 1)
{
    File delName = new File("st.txt");
    File oldName = new File("st1.txt");
    File newName = new File("st.txt");

    if(delName.delete())
```

```
{
   System.out.println("Record deleted successfully.");
}

else
{
   System.out.println("Error.");
}

if(oldName.renameTo(newName))
{
pw.close();
   if(flag == 1)
   {
      File del_Name = new File("st.txt");
      File old_Name = new File("st1.txt");
      File new_Name = new File("st.txt");

      if(del_Name.delete())
      {
         System.out.println("Record deleted successfully.");
      }

      else
      {
         System.out.println("Error.");
      }

      if(old_Name.renameTo(newName))
      {
         System.out.println("Renamed successfully.");
```

```
                }


                else
                {
                   System.out.println("Error.");
                }
             }


             else
             {
                System.out.println("Record not found.");
             }
          }
       }
    }
    catch(Exception e)
    {
       System.out.println(e);
    }
}


static void updateRecord()
{
    try
    {
       BufferedReader file1 = new BufferedReader(new FileReader("st.txt"));
       PrintWriter pw = new PrintWriter(new BufferedWriter(new FileWriter("st1.txt")));


       String name;
       int flag = 0;
```

```
Scanner sc = new Scanner(System.in);

System.out.print("Enter the name of the student you want to update: ");
String searchName = sc.nextLine();

while((name = file1.readLine())!=null)
{
   String[]line = name.split(" ");
   if(!searchName.equalsIgnoreCase(line[0]))
   {
      pw.println(name);
      flag = 0;
   }

   else
   {
      System.out.println("Record found.");
      System.out.println("Enter updated marks: ");

      float up_mark = sc.nextFloat();
      pw.println(line[0]+" "+line[1]+" "+line[2]+" "+line[3]+" "+line[4]+" "+up_mark);
      flag = 1;

      while((name = file1.readLine())!=null)
      {
         pw.println(name);
      }
   break;
   }
}
```

```
file1.close();
pw.close();
if(flag == 1)
{
   File delName = new File("st.txt");
   File oldName = new File("st1.txt");
   File newName = new File("st.txt");

   if(delName.delete())
   {
      System.out.println("Record updated successfully.");
   }

   else
   {
      System.out.println("Error.");
   }

   if(oldName.renameTo(newName))
   {
      System.out.println("Renamed successfully.");
   }
   else
   {
      System.out.println("Error.");
   }
}

else
{
   System.out.println("Record not found.");
```

```
        }
    }
    catch(Exception e)
    {
        System.out.println(e);
    }
}

static void searchRecords()
{
    try
    {
        BufferedReader file = new BufferedReader(new FileReader("st.txt"));
        String name;
        int flag = 0;
        Scanner sc = new Scanner(System.in);

        System.out.print("Enter the id of the student you want to search: ");
        String searchName = sc.nextLine();

        while((name = file.readLine())!= null)
        {
            String[]line = name.split(" ");
            if(searchName.equalsIgnoreCase(line[1]))
            {
                System.out.println("Record found.");
                System.out.println(name);
                System.out.println();

                flag = 1;
                break;
```

```
            }
        }

        if(flag == 0)
        {
            System.out.println("Record not found.");
        }

        file.close();
    }
    catch(Exception e)
    {
        System.out.println(e);
    }
}
static void displayRecords()
{
    try
    {
        BufferedReader file = new BufferedReader(new FileReader("st.txt"));
        String name;
        int i = 1;

        while((name = file.readLine())!= null)
        {
            System.out.println(name);
            System.out.println("\n");
        }

        file.close();
    }
```

```
    catch(Exception e)
    {
        System.out.println(e);
    }
}

static void clearRecords()
{
    try
    {
        File delName = new File("st.txt");
        if(delName.delete())
        {
            System.out.println("Cleared records.");
        }

        else
        {
            System.out.println("Error.");
        }

        PrintWriter pw = new PrintWriter(new BufferedWriter(new FileWriter("st.txt")));
        pw.close();
        System.out.println("All records are cleared successfully.");
    }
    catch(Exception e)
    {
        System.out.println(e);
    }
}
```

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


// Output:
/*

1 - Add Record
2 - Update Record
3 - Display Record
4 - Search Record
5 - Delete Record
6 - Clear All Records
7 - Exit

1


Enter student's name: Yogesh

Enter student's id: 9

Enter student's roll number: 27009

Enter address: Moshi

Enter student's class: 12
```

Enter marks: 88

Record added successfully.

Do you want to add more?(Y/N): Y

Enter student's name: Gaurav

Enter student's id: 10

Enter student's roll number: 27010

Enter address: Bhosari

Enter student's class: 12

Enter marks: 85

Record added successfully.

Do you want to add more?(Y/N): Y

Enter student's name: Abhay

Enter student's id: 15

Enter student's roll number: 27015

Enter address: Pune

Enter student's class: 12

Enter marks: 80

Record added successfully.

Do you want to add more?(Y/N): N

1 - Add Record

2 - Update Record

3 - Display Record

4 - Search Record

5 - Delete Record

6 - Clear All Records

7 - Exit

2

Enter the name of the student you want to update: Abhay

Record found.

Enter updated marks:

84

Record updated successfully.

Renamed successfully.

1 - Add Record

2 - Update Record

3 - Display Record

4 - Search Record

5 - Delete Record

6 - Clear All Records

7 - Exit

3

Yogesh 9 27009 Moshi 12 88.0

Gaurav 10 27010 Bhosari 12 85.0

Abhay 15 27015 Pune 12 84.0

1 - Add Record

2 - Update Record

3 - Display Record

4 - Search Record

5 - Delete Record

6 - Clear All Records

7 - Exit

4

Enter the id of the student you want to search: 15

Record found.

Abhay 15 27015 Pune 12 84.0

1 - Add Record

2 - Update Record

3 - Display Record

4 - Search Record

5 - Delete Record

6 - Clear All Records

7 - Exit

5

Enter the name of the student you want to delete: Abhay

Record deleted successfully.

Record deleted successfully.

1 - Add Record

2 - Update Record

3 - Display Record

4 - Search Record

5 - Delete Record

6 - Clear All Records

7 - Exit

6

All records are cleared successfully.

1 - Add Record

2 - Update Record

3 - Display Record

4 - Search Record

5 - Delete Record

6 - Clear All Records

7 - Exit

7

*/

Name

More sorting options

SE- OOPL Syllabus.pdf

More actions (Alt+A)

SE OOPL Viva Questions For Lab Assignment-7 To 11.pdf

More actions (Alt+A)

ASSIGNMENT_11.java

More actions (Alt+A)

ASSIGNMENT_10.java

More actions (Alt+A)

ASSIGNMENT_9.java

More actions (Alt+A)

ASSIGNMENT_8.java

More actions (Alt+A)

ASSIGNMENT_7.java

More actions (Alt+A)

/*

Assignment No.9

Title:- Case Study.

Problem Statement:- Using concepts of Object-Oriented programming develop solution for any one application:

1) Banking system having following operations:

1. Create an account 2. Deposit money 3. Withdraw money 4. Honor daily withdrawal limit 5. Check the balance 6. Display Account information.

2) Inventory management system having following operations:

1. List of all products 2. Display individual product information 3. Purchase 4. Shipping  5. Balance stock 6. Loss and Profit calculation.

*/

// CODE

/* We will implement following case study:

1) Banking system having following operations:

1. Create an account 2. Deposit money 3. Withdraw money 4. Honor daily withdrawal limit 5. Check the balance 6. Display Account information.

*/

// FILE NAME: Assignment_9.java

```
import java.util.*;
import java.io.*;

public class Assignment_9
{
   public static void main(String[]args)
   {
      int op;
      Scanner sc = new Scanner(System.in);
      Bank b1 = new Bank();
      while(true)
      {
         System.out.println();
         System.out.print("1 - Create Account\n2 - Display Account\n3 - Check Balance\n4 - Deposit Amount\n5 - Withdraw Amount\n6 - Exit\n");
```

```
        System.out.println("Enter Your Choice: ");
        op = sc.nextInt();
        System.out.println();

        switch(op)
        {
           case 1: b1.createAccount();
           break;

           case 2: b1.displayInfo();
           break;

           case 3: b1.checkBalance();
           break;

           case 4: b1.depositAmount();
           break;

           case 5: b1.withdrawAmount();
           break;

           case 6: System.exit(0);
           break;

           default: System.out.println("Invalid input.");
        }
      }
   }
}
```

```
// FILE NAME: Bank.java
import java.io.*;
import java.util.*;

class Bank
{
   Scanner sc = new Scanner(System.in);
   Customer c1 = new Customer();
   SavingsAccount s1 = new SavingsAccount();
   Account a1 = new Account();

   void createAccount()
   {
      c1.setCustomerName();
      c1.setCustomerAge();
      a1.setAccId();
      a1.setAccountType();
      a1.setBalance();
      s1.setMinimumBalance();
   }


   void withdrawAmount()
   {
      int withdraw;
      System.out.print("Enter the amount you want to withdraw: ");
      withdraw = sc.nextInt();

      if(a1.getBalance() - withdraw < s1.getMinimumBalance())
      {
```

```
            System.out.println("Withdrawal Failed. Not enough balance.");
        }

        else if(withdraw > s1.getMinimumBalance())
        {
            System.out.println("Withdrawal Failed. Maximum limit of one transaction is
"+s1.getMinimumBalance()+".");
        }

        else
        {
            a1.balance = a1.getBalance() - withdraw;
        }
    }

    void displayInfo()
    {
        String name = c1.getCustomerName();
        int age = c1.getCustomerAge();
        int accId = a1.getAccId();
        String accType = a1.getAccountType();
        int balance = a1.getBalance();
        int minBalance = s1.getMinimumBalance();
        System.out.println("Welcome, "+name+"! Below are your account details:");
        System.out.println("Age: "+age);
        System.out.println("Account Id: "+accId);
        System.out.println("Account Type: "+accType);
        System.out.println("Balance: "+balance);
        System.out.println("Minimum Balance: "+minBalance);
    }
```

```
    void depositAmount()
    {
        int deposit;
        System.out.print("Enter the amount you want to deposit: ");
        deposit = sc.nextInt();
        a1.balance = deposit + a1.getBalance();
    }

    void checkBalance()
    {
        int balance;
        balance = a1.getBalance();
        System.out.println("Balance: "+balance);
    }

}




// FILE NAME: Account.java

import java.io.*;
import java.util.*;

class Account
{
    Scanner sc = new Scanner(System.in);
    String accType;
    int accId;
    int balance;
```

```
void setAccId()
{
   System.out.print("Enter Account Id: ");
   accId = sc.nextInt();
}

void setBalance()
{
   System.out.print("Enter balance: ");
   balance = sc.nextInt();
}

void setAccountType()
{
   System.out.print("Enter your account type: ");
   accType = sc.nextLine();
   accType = sc.nextLine();
}

public int getBalance()
{
   return balance;
}

public int getAccId()
{
   return accId;
}

public String getAccountType()
{
```

```
        return accType;
    }
}

// FILE NAME: Customer.java

import java.io.*;
import java.util.*;

class Customer
{
    Scanner sc = new Scanner(System.in);
    String name;
    int age;

    public String getCustomerName()
    {
        return name;
    }

    void setCustomerName()
    {
        System.out.print("Enter name: ");
        name = sc.nextLine();
    }

    public int getCustomerAge()
    {
        return age;
    }
```

```
    void setCustomerAge()
    {
        try
        {
            System.out.print("Minimum age for creating an account is 18.\nEnter age: ");
            age = sc.nextInt();
            if(age<18)
            {
                throw new Exception("You are not old enough to create an account");
            }
        }
        catch(Exception e)
        {
            System.out.println(e);
            System.exit(0);
        }
    }
}



// FILE NAME: SavingsAccount.java

import java.io.*;
import java.util.*;

class SavingsAccount
{
    Scanner sc = new Scanner(System.in);
    int minBalance;

    void setMinimumBalance()
```

```
    {
        System.out.print("Enter minimum balance: ");
        minBalance = sc.nextInt();
    }

    public int getMinimumBalance()
    {
        return minBalance;
    }
}




//Output:

/*


1 - Create Account
2 - Display Account
3 - Check Balance
4 - Deposit Amount
5 - Withdraw Amount
6 - Exit
Enter Your Choice:
1

Enter name: Yogesh
Minimum age for creating an account is 18.
Enter age: 19
Enter Account Id: 9
```

Enter your account type: Savings

Enter balance: 8000

Enter minimum balance: 1000

1 - Create Account

2 - Display Account

3 - Check Balance

4 - Deposit Amount

5 - Withdraw Amount

6 - Exit

Enter Your Choice:

2

Welcome, Yogesh! Below are your account details:

Age: 19

Account Id: 9

Account Type: Savings

Balance: 8000

Minimum Balance: 1000

1 - Create Account

2 - Display Account

3 - Check Balance

4 - Deposit Amount

5 - Withdraw Amount

6 - Exit

Enter Your Choice:

3

Balance: 8000

1 - Create Account

2 - Display Account

3 - Check Balance

4 - Deposit Amount

5 - Withdraw Amount

6 - Exit

Enter Your Choice:

4

Enter the amount you want to deposit: 2000

1 - Create Account

2 - Display Account

3 - Check Balance

4 - Deposit Amount

5 - Withdraw Amount

6 - Exit

Enter Your Choice:

3

Balance: 10000

1 - Create Account

2 - Display Account

3 - Check Balance

4 - Deposit Amount

5 - Withdraw Amount

6 - Exit

Enter Your Choice:

5

Enter the amount you want to withdraw: 5000

Withdrawal Failed. Maximum limit of one transaction is 1000.

1 - Create Account

2 - Display Account

3 - Check Balance

4 - Deposit Amount

5 - Withdraw Amount

6 - Exit

Enter Your Choice:

5

Enter the amount you want to withdraw: 1000

1 - Create Account

2 - Display Account

3 - Check Balance

4 - Deposit Amount

5 - Withdraw Amount

6 - Exit

Enter Your Choice:

3

Balance: 9000

1 - Create Account

2 - Display Account

3 - Check Balance

4 - Deposit Amount

5 - Withdraw Amount

6 - Exit

Enter Your Choice:

2

Welcome, Yogesh! Below are your account details:

Age: 19

Account Id: 9

Account Type: Savings

Balance: 9000

Minimum Balance: 1000

1 - Create Account

2 - Display Account

3 - Check Balance

4 - Deposit Amount

5 - Withdraw Amount

6 - Exit

Enter Your Choice:

3

Balance: 9000

1 - Create Account

2 - Display Account

3 - Check Balance

4 - Deposit Amount

5 - Withdraw Amount

6 - Exit

Enter Your Choice:

6

*/

Name

More sorting options

SE- OOPL Syllabus.pdf

More actions (Alt+A)

SE OOPL Viva Questions For Lab Assignment-7 To 11.pdf

More actions (Alt+A)

ASSIGNMENT_11.java

More actions (Alt+A)

ASSIGNMENT_10.java

More actions (Alt+A)

ASSIGNMENT_9.java

More actions (Alt+A)

ASSIGNMENT_8.java

More actions (Alt+A)

ASSIGNMENT_7.java

More actions (Alt+A)

/*

Assignment No.10

Title:- Factory Design pattern.

Problem Statement:- Design and implement Factory design pattern for the given context. Consider Car building process, which requires many steps from allocating accessories to final makeup. These steps should be written as methods and should be called while creating an instance of a specific car type. Hatchback, Sedan, SUV could be the subclasses of Car class. Car class and its subclasses, CarFactory and TestFactoryPattern should be implemented.

```
*/


//CODE

// FILE NAME: Car.java

abstract class Car
{
   private CarType model=null;

   public Car(CarType model)
   {
      this.model = model;
   }

   protected abstract void construct();   //abstract method

}


// FILE NAME: CarFactory.java

class CarFactory
{
   public static Car buildCar(CarType model)
   {
      Car car = null;
      switch (model)
      {
```

```
            case SMALL:
            car = new SmallCar();
            break;

            case SEDAN:
            car = new SedanCar();
            break;

            case LUXURY:
            car = new LuxuryCar();
            break;

            default:

            break;
        }
            return car;
    }
}



// FILE NAME: CarType.java

enum CarType
{
    SMALL, SEDAN, LUXURY
}



// FILE NAME: LuxuryCar.java
```

```
class LuxuryCar extends Car
{
   LuxuryCar()
   {
      super(CarType.LUXURY);
      construct();
   }

   protected void construct()
   {
      System.out.println("Building LUXURY car");
   }
}


// FILE NAME: SedanCar.java

class SedanCar extends Car
{
   SedanCar()
   {
      super(CarType.SEDAN);
      construct();
   }

   protected void construct()
   {
      System.out.println("Building SEDAN car");

   }
}
```

```
// FILE NAME: SmallCar.java

class SmallCar extends Car
{
   SmallCar()
   {
      super(CarType.SMALL);
      construct();
   }

   protected void construct()
   {
      System.out.println("Building small car");
   }
}



// FILE NAME: Assignment_10.java

public class Assignment_10
{
   public static void main(String[] args)
   {
      System.out.println(CarFactory.buildCar(CarType.SMALL));
      System.out.println(CarFactory.buildCar(CarType.SEDAN));
      System.out.println(CarFactory.buildCar(CarType.LUXURY));
   }
}
```

// Output:

/*

Building small car

assignment10.SmallCar@19f2327

Building SEDAN car

assignment10.SedanCar@bedef2

Building LUXURY car

assignment10.LuxuryCar@7716f4

*/

More actions (Alt+A)

ASSIGNMENT_8.java

More actions (Alt+A)

ASSIGNMENT_7.java

More actions (Alt+A)

```
/*
Assignment No.11
Title:- Strategy Design Pattern.

Problem Statement:- Implement and apply Strategy Design pattern for simple Shopping Cart where three payment strategies are used such as Credit Card, PayPal, BitCoin. Create the interface for strategy pattern and give concrete implementation for payment.

*/

//CODE

// FILE NAME: PaymentStrategy.java

public interface PaymentStrategy
{
   public void pay(int amount);
}



// FILE NAME: ShoppingCart.java

import java.util.ArrayList;
import java.util.List;

public class ShoppingCart
```

```
{
    List<Item> items;

    public ShoppingCart()
    {
        this.items=new ArrayList<Item>();
    }

    public void addItem(Item item)
    {
        this.items.add(item);
    }

    public void removeItem(Item item)
    {
        this.items.remove(item);
    }

    public int calculateTotal()
    {
        int sum = 0;
        for(Item item : items)
        {
            sum += item.getPrice();
        }

        return sum;
    }

    public void pay(PaymentStrategy paymentMethod)
    {
```

```
        int amount = calculateTotal();
        paymentMethod.pay(amount);
    }
}


// FILE NAME: Item.java

public class Item
{
    private String upcCode;
    private int price;

    public Item(String upc, int cost)
    {
        this.upcCode=upc;
        this.price=cost;
    }

    public String getUpcCode()
    {
        return upcCode;
    }

    public int getPrice()
    {
        return price;
    }
}
```

```
// FILE NAME: PaypalStrategy.java

public class PaypalStrategy implements PaymentStrategy
{
   private String emailId;
   private String password;

   public PaypalStrategy(String email, String pwd)
   {
      this.emailId=email;
      this.password=pwd;
   }

   @Override
   public void pay(int amount)
   {
      System.out.println(amount + " paid using Paypal.");
   }
}




// FILE NAME: BitCoinStrategy.java

public class BitCoinStrategy implements PaymentStrategy
{
   private String recipient_bitcoin_address;

   public BitCoinStrategy(String recipient_bitcoin_address)
   {
      this.recipient_bitcoin_address=recipient_bitcoin_address;
```

```
    }

    @Override
    public void pay(int amount)
    {
        System.out.println(amount + " paid using Bitcoin.");
    }
}



// FILE NAME: CreditCardStrategy.java

public class CreditCardStrategy implements PaymentStrategy
{
    private String name;
    private String cardNumber;
    private String cvv;
    private String dateOfExpiry;

    public CreditCardStrategy(String nm, String ccNum, String cvv, String expiryDate)
    {
    this.name=nm;
    this.cardNumber=ccNum;
    this.cvv=cvv;
    this.dateOfExpiry=expiryDate;
    }

    @Override
    public void pay(int amount)
    {
        System.out.println(amount +" paid with credit/debit card.");
```

```
    }
}



// FILE NAME: Assignment_11.java

public class Assignment_11
{
    public static void main(String[] args)
    {
        ShoppingCart cart = new ShoppingCart();
        Item item1 = new Item("Plate",60);
        Item item2 = new Item("Glasses",120);
        Item item3 = new Item("Rice 5 kg",250);
        Item item4 = new Item("Oil 15L Pack",1400);
        Item item5 = new Item("Shoes",250);
        Item item6 = new Item("Men's Shirt",400);
        Item item7 = new Item("Tea Powder",80);
        Item item8 = new Item("Coffee Powder",100);

        cart.addItem(item1);
        cart.addItem(item2);
        cart.addItem(item3);
        cart.addItem(item4);
        cart.addItem(item5);
        cart.addItem(item6);
        cart.addItem(item7);
        cart.addItem(item8);

        cart.pay(new PaypalStrategy("yogeshborhade24@gmail.com", "PaSsWoRd"));
```

```
        cart.pay(new CreditCardStrategy("Yogesh Borhade", "12365896574", "125", "11/24"));
        cart.pay(new BitCoinStrategy("shoppingmall@bitcoin.sbi"));
    }

}

// Output:

/*

2660 paid using Paypal.
2660 paid with credit/debit card.
2660 paid using Bitcoin.

*/
```