**NAME -P.HARI KIRAN**

**ID NUMBER -170031577**

**MAJOR PROJECTS**

**1.CREDIT CARD BILL**

**CreditCardBill:**

package com.wipro.ccbill.entity;

import java.util.Date;

import com.wipro.ccbill.exception.InputValidationException;

public class CreditCardBill {

private String creditCardNo;

private String customerId;

private Date billDate;

private Transaction monthTransactions[];

public CreditCardBill (){

super();

}

public CreditCardBill (String creditCardNo, String customerId, Date BillDate, Transaction monthTransactions[]){

this.creditCardNo = creditCardNo;

this.customerId = customerId;

this.billDate = billDate;

this.monthTransactions = monthTransactions;

}

public double getTotalAmount(String transactionType){

double amount=0.0;

if(! (transactionType.equals("DB") || transactionType.equals("CR")) )

return 0.0;

else{

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

if(monthTransactions[i].getTransactionType().equals(transactionType))

amount += monthTransactions[i].getTransactionAmount();

}

return amount;

}

}

public double calculateBillAmount(){

try{

if(validateData().equals("valid")){

if(monthTransactions!=null && monthTransactions.length >0){

double amountSpend = getTotalAmount("DB");

double amountpaid = getTotalAmount("CR");

double outstanding = amountSpend - amountpaid;

double interest = (outstanding\*(19.9/100))/12;

return outstanding + interest;

} else{

return 0.0;

}

}

} catch(InputValidationException ive){

return 0.0;

}

return 0.0;

}

public String validateData() throws InputValidationException{

if(creditCardNo == null

|| creditCardNo.length() !=16 || customerId == null || customerId.length() <6 )

throw new InputValidationException();

return "valid";

}

}

**Transaction:**

package com.wipro.ccbill.entity;

import java.util.Date;

public class Transaction {

private String creditCardNo;

private Date dateOfTransaction;

private String transactionDesc;

private double transactionAmount;

private String transactionType;

// No args Constructor

public Transaction (){

super();

}

public Transaction (String creditCardNo, Date dateOfTransaction, String transactionDesc, double transactionAmount, String transactionType){

this.creditCardNo = creditCardNo;

this.dateOfTransaction = dateOfTransaction;

this.transactionDesc = transactionDesc;

this.transactionAmount = transactionAmount;

this.transactionType = transactionType;

}

public String getCreditCardNo() {

return creditCardNo;

}

public void setCreditCardNo(String creditCardNo) {

this.creditCardNo = creditCardNo;

}

public Date getDateOfTransaction() {

return dateOfTransaction;

}

public void setDateOfTransaction(Date dateOfTransaction) {

this.dateOfTransaction = dateOfTransaction;

}

public String getTransactionDesc() {

return transactionDesc;

}

public void setTransactionDesc(String transactionDesc) {

this.transactionDesc = transactionDesc;

}

public double getTransactionAmount() {

return transactionAmount;

}

public void setTransactionAmount(double transactionAmount) {

this.transactionAmount = transactionAmount;

}

public String getTransactionType() {

return transactionType;

}

public void setTransactionType(String transactionType) {

this.transactionType = transactionType;

}

}

InputValidationException:

package com.wipro.ccbill.exception;

public class InputValidationException extends Exception{

public InputValidationException(){

super();

}

public InputValidationException(String err){

super(err);

}

@Override

public String toString() {

return "Invalid Input Data";

}

}

**MainClass:**

package com.wipro.ccbill.main;

import java.text.SimpleDateFormat;

import com.wipro.ccbill.entity.CreditCardBill;

import com.wipro.ccbill.entity.Transaction;

public class MainClass {

public static void main(String[] args) {

try{

Transaction monthsTransaction[] = new Transaction[5];

SimpleDateFormat formatter = new SimpleDateFormat("dd/MM/yyyy");

monthsTransaction[0]=new Transaction("1111222233334444",formatter.parse("01/02/2016"),"household",8000.0,"DB");

monthsTransaction[1]=new Transaction("1111222233334444",formatter.parse("05/02/2016"),"Textile",12000.0,"DB");

monthsTransaction[2]=new Transaction("1111222233334444",formatter.parse("12/02/2016"),"movie",2700.0,"DB");

monthsTransaction[3]=new Transaction("1111222233334444",formatter.parse("16/02/2016"),"paid",19000.0,"CR");

monthsTransaction[4]=new Transaction("1111222233334444",formatter.parse("19/02/2016"),"household",4500.0,"DB");

CreditCardBill ccbill = new CreditCardBill("1111222233334444","ABC123",formatter.parse("02/03/2016"),monthsTransaction);

System.out.println("Total to be paid : "+ccbill.calculateBillAmount());

}catch(Exception e){

}

}

}

**2.INTEREST CALCULATION**

**Account:**

package com.wipro.bank.acc;

public abstract class Account {

int tenure;

float principal;

float rateOfInterest;

public void setInterest(int age, String gender)

{

if(gender.equalsIgnoreCase("male"))

{

if(age<60)

{

this.rateOfInterest=9.8f;

}

else

{

this.rateOfInterest=10.5f;

}

}

if(gender.equalsIgnoreCase("female"))

{

if(age<58)

{

this.rateOfInterest=10.2f;

}

else

{

this.rateOfInterest=10.8f;

}

}

}

public float calculateMaturityAmount(float totalPrincipleDeposited,float maturityInterest)

{

return totalPrincipleDeposited+maturityInterest;

}

public abstract float calculateInterest();

public abstract float calculateAmountDeposited();

public int getTenure() {

return tenure;

}

public void setTenure(int tenure) {

this.tenure = tenure;

}

public float getPrincipal() {

return principal;

}

public void setPrincipal(float principal) {

this.principal = principal;

}

public float getRateOfInterest() {

return rateOfInterest;

}

public void setRateOfInterest(float rateOfInterest) {

this.rateOfInterest = rateOfInterest;

}

}

RDAccount:

package com.wipro.bank.acc;

public class RDAccount extends Account{

public RDAccount(int tenure, float principal)

{

this.tenure=tenure;

this.principal=principal;

}

public float calculateAmountDeposited()

{

System.*out*.println("Amount Deposited");

return principal\*tenure\*12;

}

public float calculateInterest()

{

float p=(1+rateOfInterest/4/100);

float n=(float)Math.*pow*(p,4\*tenure);

float interest=principal\*(n-1);

System.*out*.println("Interest:");

return interest;

}

}

BankValidationException:

package com.wipro.bank.exception;

public class BankValidationException extends Exception {

/\*\*

\*

\*/

private static final long *serialVersionUID* = 1L;

public String toString()

{

return "invalid data";

}

}

**BankService:**

package com.wipro.bank.service;

import com.wipro.bank.acc.\*;

import com.wipro.bank.exception.BankValidationException;

public class BankService {

public boolean validateData(float principal, int tenure,int age, String gender)

{

boolean value=true;

try

{

if(principal<500 || (tenure!=5 && tenure!=10) || ((!gender.equalsIgnoreCase("male")&&(!gender.equalsIgnoreCase("female"))) || (age<1) || (age>100)))

{

throw new BankValidationException();

}

value=true;

}

catch(BankValidationException e)

{

System.out.println(e);

}

return value;

}

public void calculate(float principal,int tenure, int age, String gender)

{

if(validateData(principal,tenure,age,gender)==true)

{

Account ac=new RDAccount(tenure,principal);

ac.setInterest(age,gender);

System.out.println(ac.calculateInterest());

System.out.println(ac.calculateAmountDeposited());

System.out.println(ac.calculateMaturityAmount(principal,ac.getRateOfInterest()));

}

}

}

**Main:**

**package** com.wipro.bank.main;

import com.wipro.bank.service.BankService;

public class Main {

public static void main(String[] args) {

// TODO Auto-generated method stub

int tenure = 5;

float principal = 1000;

int age = 23;

String gender = "Male";

BankService b=new BankService();

b.calculate(principal, tenure, age, gender);

}

}

**3.ElectricityBill:**

**Commercial:**

package com.wipro.eb.entity;

public class Commercial extends Connection

{

public Commercial(int previousReading, int currentReading, float[] slabs)

{

super(previousReading, currentReading, slabs);

// TODO Auto-generated constructor stub

}

@Override

public float computeBill()

{

int units=currentReading-previousReading;

float billamount;

if(units<50)

{

billamount=units\*slabs[0];

}

else if(units<100)

{

billamount=(float) ((50\*slabs[0])+((units-50)\*slabs[1]));

}

else

{

billamount=(float) ((50\*slabs[0])+(50\*slabs[1])+((units-100)\*slabs[2]));

}

float electricityduty=0.0f;

if(billamount>=10000)

{

electricityduty=(float)(billamount\*0.09);

}

else if(billamount>=5000)

{

electricityduty=(float)(billamount\*0.06);

}

else

{

electricityduty=(float) (billamount\*0.02);

}

billamount+=electricityduty;

return billamount;

}

}

**Connection:**

package com.wipro.eb.entity;

public abstract class Connection

{

int previousReading;

int currentReading;

float[] slabs;

public Connection(int previousReading, int currentReading, float[] slabs)

{

super();

this.previousReading = previousReading;

this.currentReading = currentReading;

this.slabs = slabs;

}

public abstract float computeBill();

public int getPreviousReading()

{

return previousReading;

}

public void setPreviousReading(int previousReading)

{

this.previousReading = previousReading;

}

public int getCurrentReading()

{

return currentReading;

}

public void setCurrentReading(int currentReading)

{

this.currentReading = currentReading;

}

public float[] getSlabs()

{

return slabs;

}

public void setSlabs(float[] slabs)

{

this.slabs = slabs;

}

}

**Domestic:**

package com.wipro.eb.entity;

public class Domestic extends Connection

{

public Domestic(int previousReading, int currentReading, float[] slabs)

{

super(previousReading, currentReading, slabs);

// TODO Auto-generated constructor stub

}

@Override

public float computeBill()

{

float billamount;

int units=currentReading-previousReading;

if(units<50)

{

billamount=units\*slabs[0];

}

else if(units<100)

{

billamount=(float) (50\*2.3+(units-50)\*slabs[1]);

}

else

{

billamount=(float) ((50\*slabs[0])+(50\*slabs[1])+((units-100)\*slabs[2]));

}

// TODO Auto-generated method stub

return billamount;

}

}

**InvalidConnectionException:**

package com.wipro.eb.exception;

public class InvalidConnectionException extends Exception

{

public String toString()

{

return "Invalid Connection Type";

}

}

**InvalidReadingException:**

package com.wipro.eb.exception;

public class InvalidReadingException extends Exception

{

public String toString()

{

return "Incorrect Reading";

}

}

**ConnectionService:**

package com.wipro.eb.service;

import com.wipro.eb.entity.Commercial;

import com.wipro.eb.entity.Domestic;

import com.wipro.eb.exception.InvalidConnectionException;

import com.wipro.eb.exception.InvalidReadingException;

public class ConnectionService

{

public boolean validate(int currentReading,int previousReading,String type)throws InvalidReadingException,InvalidConnectionException

{

boolean b=true;

if((currentReading<previousReading)||(currentReading<0)||(previousReading<0))

{

b=false;

throw new InvalidReadingException();

}

else if(!((type.equalsIgnoreCase("Domestic"))||type.equalsIgnoreCase("Commercial")))

{

b=false;

throw new InvalidConnectionException();

}

return b;

}

public float calculateBillAmt(int currentReading, int previousReading, String type) throws InvalidReadingException, InvalidConnectionException

{

float bill = 0;

try

{

if(validate(currentReading,previousReading,type))

{

if(type.equalsIgnoreCase("Commercial"))

{

float[] slabs = {5.2f, 6.8f, 8.3f};

Commercial c=new Commercial(previousReading,currentReading,slabs);

bill=c.computeBill();

}

else if(type.equalsIgnoreCase("Domestic"))

{

float[] slabs = {2.3f, 4.2f, 5.5f};

Domestic d=new Domestic(previousReading,currentReading,slabs);

bill=d.computeBill();

}

}

}

catch(InvalidReadingException e1)

{

return -1;

}

catch(InvalidConnectionException e2)

{

return -2;

}

return bill;

}

public String generateBill(int currentReading, int previousReading, String type) throws InvalidReadingException, InvalidConnectionException

{

float a=calculateBillAmt(currentReading,previousReading,type);

if(a==-1f)

{

return "Incorrect Reading";

}

else if(a==-2f)

{

return "Invalid ConnectionType";

}

else

{

return("Amount to be paid is "+a);

}

}

}

**EBMain:**

package com.wipro.eb.main;

import com.wipro.eb.exception.InvalidConnectionException;

import com.wipro.eb.exception.InvalidReadingException;

import com.wipro.eb.service.ConnectionService;

public class EBMain

{

public static void main(String[] args) throws InvalidReadingException, InvalidConnectionException

{

ConnectionService c = new ConnectionService();

// Domestic Testing

System.out.println(c.generateBill(130, 120, "Domestic"));

//Output bust be 23.0

System.out.println(c.generateBill(180, 120, "Domestic"));

//Output bust be 157.0

System.out.println(c.generateBill(280, 120, "Domestic"));

//Output bust be 655.0

System.out.println(c.generateBill(-280, 120, "Domestic"));

//Output bust be Incorrect Reading

System.out.println(c.generateBill(280, -120, "Domestic"));

//Output bust be Incorrect Reading

System.out.println(c.generateBill(100, 120, "Domestic"));

//Output bust be Incorrect Reading

System.out.println(c.generateBill(200, 120, "domestic"));

//Output bust be Invalid ConnectionType

System.out.println(c.generateBill(200, 120, "dom"));

//Output bust be Invalid ConnectionType

//Commercial Testing

System.out.println(c.generateBill(130, 120, "Commercial"));

//Output bust be 53.04

System.out.println(c.generateBill(180, 120, "Commercial"));

//Output bust be 334.56

System.out.println(c.generateBill(280, 120, "Commercial"));

//Output bust be 1119.96

System.out.println(c.generateBill(-280, 120, "Commercial"));

//Output bust be Incorrect Reading

System.out.println(c.generateBill(280, -120, "Commercial"));

//Output bust be Incorrect Reading

System.out.println(c.generateBill(100, 120, "Commercial"));

//Output bust be Incorrect Reading

System.out.println(c.generateBill(200, 120, "commercial"));

//Output bust be Invalid ConnectionType

System.out.println(c.generateBill(200, 120, "comm"));

//Output bust be Invalid ConnectionType

}

}

**MINI PROJECTS**

1.**JAVA FUNDAMENTAL PROJECT:**

import java.util.Scanner;

public class Project1

{

public static void main(String[] args)

{

int[] empid=new int[] {1001,1002,1003,1004,1005,1006,1007};

String[] empname=new String[] {"Ashish","Sushma","Rahul","Chahat","Ranjan","Suman","Tanmay"};

String[] joindate=new String[] {"01/04/2009","23/08/2012","12/11/2008","29/01/2003","16/07/2005","1/1/2000","12/06/2006"};

char[] designationcode=new char[] {'e','c','k','r','m','e','c'};

String[] dept=new String[] {"R&D","PM","Acct","Front Desk","Engg","Manufacturing","PM"};

int[] basic=new int[] {20000,30000,10000,12000,50000,23000,29000};

int[] hra=new int[] {8000,12000,8000,6000,20000,9000,12000};

int[] it=new int[] {3000,9000,1000,2000,20000,4400,10000};

int user\_query;

double salary;

int len;

Scanner sc=new Scanner(System.in);

System.out.println("Enter the Employee ID:");

user\_query=sc.nextInt();

len=empid.length;

int pos=0;

String designation=null;

int da=0;

char a;

int flag=0;

for(int i=0;i<len;i++)

{

if(user\_query==empid[i])

{

flag=0;

pos=i;

a=designationcode[i];

switch(a)

{

case 'e': designation="Engineer";

da=20000;

case 'c': designation="Consultant";

da=32000;

case 'k': designation="Clerk";

da=12000;

case 'r': designation="Receptionist";

da=15000;

case 'm': designation="Manager";

da=40000;

}

salary=basic[i]+hra[i]+da-it[i];

System.out.println("Emp No. Emp Name Department Designation Salary");

System.out.println(empid[i]+" "+empname[i]+" "+dept[i]+" "+ designation+" "+salary);

break;

}

else

{

flag=1;

}

}

if(flag==1)

{

System.out.println("There is no employee with empid: "+user\_query);

}

}

}

2.**OOPS/INHERITANCE** :

**Video:**

package bean;

public class Video {

private String videoName;

private boolean checkout;

private int rating;

public Video(String name) {

videoName = name;

}

public String getName() {

return videoName;

}

public void doCheckout() {

checkout = true;

}

public void doReturn() {

checkout = false;

}

public void receiveRating(int rating) {

this.rating = rating;

}

public int getRating() {

return rating;

}

public boolean getCheckout() {

return checkout;

}

}

**VideoStore.java**

package bean;

public class VideoStore {

private Video[] store;

public int getStoreSize() {

if (store != null) return store.length;

else return 0;

}

public Video getLastAdded() {

if (store != null) return store[store.length - 1];

else return null;

}

public void addVideo(String name) {

Video video = new Video(name);

int storeSize;

try {

storeSize = store.length;

} catch (Exception e) {

storeSize = 0;

}

Video[] newStore = new Video[storeSize + 1];

newStore[storeSize] = video;

store = newStore;

}

public void doCheckout(String name) {

if (store == null || store.length == 0) {

System.out.println("Store is empty");

return;

}

for (Video video : store) {

if (video.getName().equals(name)) {

video.doCheckout();

}

}

}

public void doReturn(String name) {

if (store == null || store.length == 0) {

System.out.println("Store is empty");

return;

}

for (Video video : store) {

if (video.getName().equals(name)) {

video.doReturn();

}

}

}

public void receiveRating(String name, int rating) {

if (store == null || store.length == 0) {

System.out.println("Store is empty");

return;

}

for (Video video : store) {

if (video.getName().equals(name)) {

video.receiveRating(rating);

}

}

}

public void listInventory() {

if (store == null || store.length == 0) {

System.out.println("Store is empty");

return;

}

for (int i = 0; i < 80; i++) System.out.print("-");

System.out.printf("\n\t%-20s\t|\t%-10s\t|\t%-15s\n", "Name", "Rating", "Checkout");

for (int i = 0; i < 80; i++) System.out.print("-");

for (Video video : store) {

System.out.printf("\n\t%-20s\t|\t%-10s\t|\t%-15s\n", video.getName(), video.getRating(), video.getCheckout());

}

for (int i = 0; i < 80; i++) System.out.print("-");

}

}

**VideoLauncher.java**

package main;

import java.util.Scanner;

import bean.VideoStore;

public class VideoLauncher {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

VideoStore store = new VideoStore();

int choice = 0;

do {

System.out.println("\n1. Add Videos: \n" +

"2. Check Out Video: \n" +

"3. Return Video: \n" +

"4. Receive Rating: \n" +

"5. List Inventory: \n" +

"6. Exit: \n" +

"Enter your choice (1..6): ");

choice = sc.hasNextInt() ? sc.nextInt() : 6;

sc.nextLine();

String name;

switch (choice) {

case 1:

System.out.println("Enter the name of the video you want to Add: ");

name = sc.nextLine();

store.addVideo(name);

System.out.println("Video " + name + " added out successfully.");

break;

case 2:

System.out.println("Enter the name of the video you want to Checkout: ");

name = sc.nextLine();

store.doCheckout(name);

System.out.println("Video " + name + " checked out successfully.");

break;

case 3:

System.out.println("Enter the name of the video you want to Return: ");

name = sc.nextLine();

store.doReturn(name);

System.out.println("Video " + name + " returned successfully.");

break;

case 4:

System.out.println("Enter the name of the video you want to Rate: ");

name = sc.nextLine();

System.out.println("Enter the rating for this video: ");

int rating = sc.nextInt();

store.receiveRating(name, rating);

System.out.println("Rating " + rating + " has been mapped to the Video " + name + ".");

break;

case 5:

store.listInventory();

break;

default:

System.out.println("Exiting...!! Thanks for using the application.");

break;

}

} while (choice != 6);

sc.close();

}

}

3.**Abstraction/Packages/Exception Handling**:

**Student.java**

package com.mile1.bean;

public class Student {

String name;

int marks[];

public Student() {

}

public Student(String name, int[] marks) {

super();

this.name = name;

this.marks = marks;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public int[] getMarks() {

return marks;

}

public void setMarks(int[] marks) {

this.marks = marks;

}

}

**NullMarksArrayException.java**

package com.mile1.exception;

public class NullMarksArrayException extends Exception {

@Override

public String toString() {

return "NullMarksArrayException occurred";

}

}

**NullNameException.java**

package com.mile1.exception;

public class NullNameException extends Exception {

@Override

public String toString() {

return "NullNameException occurred";

}

}

**NullStudentException.java**

package com.mile1.exception;

public class NullStudentException extends Exception {

@Override

public String toString() {

return "NullStudentException occurred";

}

}

**StudentReport.java**

package com.mile1.service;

import com.mile1.bean.Student;

import com.mile1.exception.NullMarksArrayException;

import com.mile1.exception.NullNameException;

import com.mile1.exception.NullStudentException;

public class StudentReport {

public String findGrade (Student studentObject) {

int[] marks = studentObject.getMarks();

int marksSum = 0;

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

if (marks[i] < 35) {

return "F";

} else {

marksSum += marks[i];

}

}

if (marksSum < 150) return "C";

else if (marksSum < 200) return "B";

else if (marksSum < 250) return "A";

else return "A+";

}

public String validate (Student studentObject)

throws NullStudentException, NullNameException, NullMarksArrayException {

if (studentObject == null) {

throw new NullStudentException();

} else {

if (studentObject.getName() == null) throw new NullNameException();

if (studentObject.getMarks() == null) throw new NullMarksArrayException();

return findGrade(studentObject);

}

}

}

**StudentService.java**

package com.mile1.service;

import com.mile1.bean.Student;

public class StudentService {

public int findNumberOfNullMarks (Student data[]) {

int nullMarksCount = 0;

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

try {

data[i].getMarks();

} catch (Exception e) {

nullMarksCount++;

}

}

return nullMarksCount;

}

public int findNumberOfNullNames (Student data[]) {

int nullNamesCount = 0;

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

try {

data[i].getName();

} catch (Exception e) {

nullNamesCount++;

}

}

return nullNamesCount;

}

public int findNumberOfNullObjects (Student data[]) {

int nullObjectsCount = 0;

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

if (data[i] == null) nullObjectsCount++;;

}

return nullObjectsCount;

}

}

**StudentMain.java**

package com.mile1.main;

import com.mile1.bean.Student;

import com.mile1.exception.NullMarksArrayException;

import com.mile1.exception.NullNameException;

import com.mile1.exception.NullStudentException;

import com.mile1.service.StudentReport;

import com.mile1.service.StudentService;

public class StudentMain {

static Student data[] = new Student[10];

static {

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

data [i]= new Student();

data [0] = new Student("A1", new int[] {72, 73, 74});

data [1] = new Student("B1", new int[] {75, 76, 77});

data [2] = new Student("C1", new int[] {99,99,99});

data [3] = new Student("C3", new int[] {100,100,99});

data [4] = new Student("B2", new int[] {13, 88, 13});

data [5] = new Student("C3", new int[] {14, 14, 99});

data [6] = new Student("A2", new int[] {77,55,12});

data [7] = new Student(null, new int[] {13, 88, 13});

data [8] = new Student("A2", null);

data [9] = null;

}

public static void main(String[] args) {

StudentReport studentReport = new StudentReport();

StudentService studentService = new StudentService();

System.out.println("Grade Calculation:");

String x = null;

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

try {

x = studentReport.validate(data[i]);

} catch (NullNameException e) {

x = "NullNameException occured";

} catch (NullMarksArrayException e) {

x = "NullMarksArrayException occured";

} catch (NullStudentException e) {

x = "NullStudentException occured";

}

System.out.println("GRADE = " + x);

}

System.out.println("Number of Objects with Marks array as null = " +

studentService.findNumberOfNullMarks(data));

System.out.println("Number of Objects with Name as null = " +

studentService.findNumberOfNullNames(data));

System.out.println("Number of Objects that are entierly null = " +

studentService.findNumberOfNullObjects(data));

System.out.println("TC7: Number of Objects with Name as null = " +

studentService.findNumberOfNullNames(data));

System.out.println("TC8: Number of Objects that are entierly null = " +

studentService.findNumberOfNullObjects(data));

System.out.println("TC9: Number of Objects with Marks array as null = " +

studentService.findNumberOfNullMarks(data));

}

}

4.**I/O Streams:**

**Employee.java**

package bean;

import java.io.Serializable;

public class Employee implements Serializable {

private int id;

private String name;

private int age;

private double salary;

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public int getAge() {

return age;

}

public void setAge(int age) {

this.age = age;

}

public double getSalary() {

return salary;

}

public void setSalary(double salary) {

this.salary = salary;

}

@Override

public String toString() {

return id + " " + name + " " + age + " " + salary;

}

}

**TestMain.java**

package service;

import java.io.FileInputStream;

import java.io.FileNotFoundException;

import java.io.FileOutputStream;

import java.io.IOException;

import java.io.ObjectInputStream;

import java.io.ObjectOutputStream;

import java.util.Scanner;

import bean.Employee;

public class TestMain {

public static void main(String[] args)

throws IOException, FileNotFoundException, ClassNotFoundException {

Scanner sc = new Scanner(System.in);

int choice = 0;

String filename = "EmployeeDB.dat";

FileOutputStream fos = new FileOutputStream(filename);

ObjectOutputStream oos = new ObjectOutputStream(fos);

FileInputStream fis = new FileInputStream(filename);

ObjectInputStream ois = new ObjectInputStream(fis);

do {

System.out.println("Main Menu");

System.out.println("1. Add an Employee");

System.out.println("2. Display All");

System.out.println("3. Exit");

System.out.println("Enter Your Choice");

choice = sc.nextInt();

switch (choice) {

case 1:

Employee emp = new Employee();

System.out.print("Enter Employee ID: ");

emp.setId(sc.nextInt());

sc.nextLine();

System.out.print("Enter Employee Name: ");

emp.setName(sc.nextLine());

System.out.print("Enter Employee Age: ");

emp.setAge(sc.nextInt());

System.out.print("Enter Employee Salary: ");

emp.setSalary(sc.nextDouble());

oos.writeObject(emp);

System.out.println("Employee Added Successfully");

break;

case 2:

System.out.println("----Report-----");

emp = null;

while (fis.available() > 0) {

emp = (Employee) ois.readObject();

System.out.println(emp);

}

System.out.println("----End of Report-----");

break;

case 3:

System.out.println("Exiting the System");

}

} while (choice != 3);

fos.close();

oos.close();

fis.close();

ois.close();

sc.close();

}

}