1) Implement a banking system using java. Create 3 sub class of Bank : SBI,BOI,ICICI

Classes should have attributes like Name, headofficeAddress, chairmanName, branchCount, fdInterestRate, personalLoanInterestRate, homeLoanInterestRate. All 3 should have following methods:

* add getters and setters for the fields
* print details of every bank (override toString)

abstract public class Bank {

private String name,headofficeAddress,chairmanName;

private int branchCount;

private double fdInterestRate,personalLoanInterestRate,homeLoanInterestRate;

public Bank(String name, String headofficeAddress, String chairmanName, int branchCount, double fdInterestRate, double personalLoanInterestRate, double homeLoanInterestRate) {

this.name = name;

this.headofficeAddress = headofficeAddress;

this.chairmanName = chairmanName;

this.branchCount = branchCount;

this.fdInterestRate = fdInterestRate;

this.personalLoanInterestRate = personalLoanInterestRate;

this.homeLoanInterestRate = homeLoanInterestRate;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public String getHeadofficeAddress() {

return headofficeAddress;

}

public void setHeadofficeAddress(String headofficeAddress) {

this.headofficeAddress = headofficeAddress;

}

public String getChairmanName() {

return chairmanName;

}

public void setChairmanName(String chairmanName) {

this.chairmanName = chairmanName;

}

public int getBranchCount() {

return branchCount;

}

public void setBranchCount(int branchCount) {

this.branchCount = branchCount;

}

public double getFdInterestRate() {

return fdInterestRate;

}

public void setFdInterestRate(double fdInterestRate) {

this.fdInterestRate = fdInterestRate;

}

public double getPersonalLoanInterestRate() {

return personalLoanInterestRate;

}

public void setPersonalLoanInterestRate(double personalLoanInterestRate) {

this.personalLoanInterestRate = personalLoanInterestRate;

}

public double getHomeLoanInterestRate() {

return homeLoanInterestRate;

}

public void setHomeLoanInterestRate(double homeLoanInterestRate) {

this.homeLoanInterestRate = homeLoanInterestRate;

}

@Override

public String toString() {

return

"name= " + name + '\n' +

"headofficeAddress= " + headofficeAddress + '\n' +

"chairmanName= " + chairmanName + '\n' +

"branchCount= " + branchCount + '\n' +

"fdInterestRate= " + fdInterestRate + '\n' +

"personalLoanInterestRate= " + personalLoanInterestRate + '\n' +

"homeLoanInterestRate= " + homeLoanInterestRate + '\n' ;

}

public static void main (String[]args){

ICICI icici = new ICICI("ICICI Bank", "Mumbai", "Sandeep Bakhshi", 5000, 5.5, 11.5, 7.5);

SBI sbi = new SBI("SBI bank","Delhi","Manish",9000,7.5,6.5, 8.8);

icici.setFdInterestRate(6.0);

icici.setPersonalLoanInterestRate(12.0);

icici.setHomeLoanInterestRate(8.0);

sbi.setFdInterestRate(5.0);

sbi.setPersonalLoanInterestRate(4.0);

sbi.setHomeLoanInterestRate(6.6);

System.*out*.println(icici.toString());

System.*out*.println(sbi.toString());

}

}

SUB-CLASSES

public class ICICI extends Bank {

public ICICI(String name, String headofficeAddress, String chairmanName, int branchCount, double fdInterestRate, double personalLoanInterestRate, double homeLoanInterestRate) {

super(name, headofficeAddress, chairmanName, branchCount, fdInterestRate, personalLoanInterestRate, homeLoanInterestRate);

}

}

public class BOI extends Bank {

public BOI(String name, String headofficeAddress, String chairmanName, int branchCount, double fdInterestRate, double personalLoanInterestRate, double homeLoanInterestRate) {

super(name, headofficeAddress, chairmanName, branchCount, fdInterestRate, personalLoanInterestRate, homeLoanInterestRate);

}

}

public class SBI extends Bank {

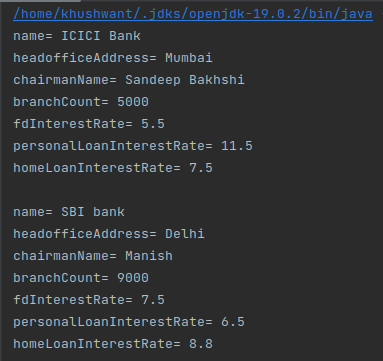
public SBI(String name, String headofficeAddress, String chairmanName, int branchCount, double fdInterestRate, double personalLoanInterestRate, double homeLoanInterestRate) {

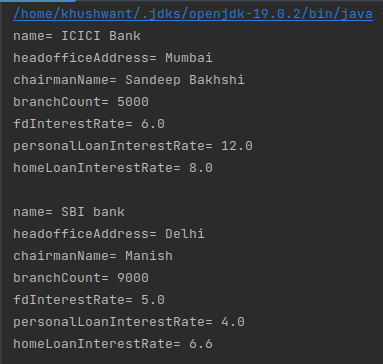
super(name, headofficeAddress, chairmanName, branchCount, fdInterestRate, personalLoanInterestRate, homeLoanInterestRate);

}

}

OUTPUT-





2) WAP showing try, multi-catch and finally blocks.

public class Example {

public static void main(String[] args) {

try {

int x = 5 / 0;

} catch (ArithmeticException e) {

System.*out*.println("Caught arithmetic exception: " + e.getMessage());

} catch (Exception e) {

System.*out*.println("Caught exception: " + e.getMessage());

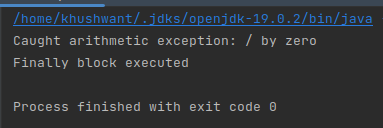
} finally {

System.*out*.println("Finally block executed");

}

}

}



3) WAP to produce NoClassDefFoundError and ClassNotFoundException exception.

public class Main {

public static void main(String[] args) {

try{

Class.*forName*("ExceptionClass");

} catch (ClassNotFoundException e) {

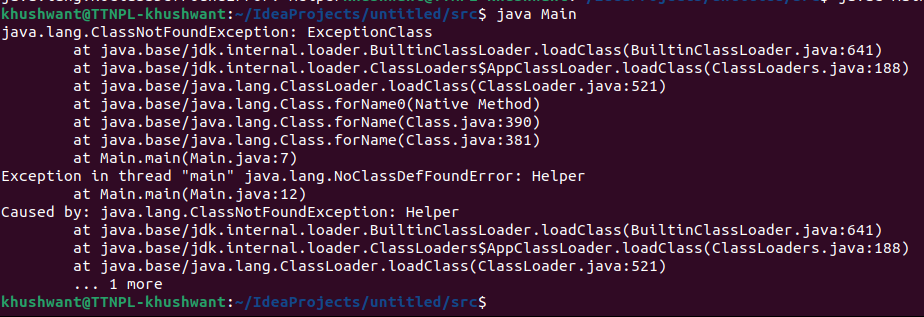
e.printStackTrace();

}

Helper help = new Helper();

}

}



4) Create a custom exception that do not have any stack trace.

class Ex extends Exception

{

public Ex(int age)

{

if(age < 18){

System.*out*.print("Below 18 not allowed ");

}else{

System.*out*.print("Allowed ");

}

}

@Override

public synchronized Throwable fillInStackTrace() {

return null;

}

}

import java.util.Scanner;

public class Main {

public static void EnterAge() throws Ex{

Scanner input = new Scanner(System.*in*);

System.*out*.print("Enter Age");

int age = input.nextInt();

throw new Ex(age);

}

public static void main(String[] args) {

try{

*EnterAge*();

} catch (Ex e) {

e.printStackTrace();

}

}

}

