**Main:**

package testaccounts;

import java.util.InputMismatchException;

import java.util.Scanner;

public class TestAccounts {

public static void main(String[] args) {

Scanner s = new Scanner(System.in); //Creating 3 Scanner objects

Scanner s1= new Scanner(System.in);

Scanner s2= new Scanner(System.in);

System.out.println(" THIS IS A SIMPLE BANK ACCOUNT PROGRAM");

System.out.print("User can open accounts,deposit and withdraw money,set his balance and email later on. ");

System.out.println("User can also review his account informations with comfort. Thank you.");

Account[] acc = new Account[20];

String email,acTitle;

int id;

String obal;

double openingBal;

System.out.println("How many Account you want to create?");

int x=s.nextInt();

for (int i = 1; i <= x; i++) { //For loop start

//try start

try{

System.out.println("Do You have email?\n1.yes\n2.No");

//System.out.println("Note: Do not press any letters here, press numerical digits to choose answer \n User Input is:");

int n=s.nextInt();

//switch case Start to Create Account

switch(n){

case 1:

System.out.println("Enter Account Title:"); /\*to pass argument to constractor with email\*/

acTitle=s1.nextLine();

System.out.println("Enter opening Balance:(minimum 5000)");

obal=s.next();

openingBal=Double.parseDouble(obal) ;

if(openingBal<5000){

throw new IllegalArgumentException();

}

System.out.println("Enter Email Address:");

email=s1.nextLine();

Account account= new Account(i,acTitle,openingBal,email);

acc[i]=account;

System.out.println(+i+" no Account created Successfully. \n");

break;

case 2:

System.out.println("Enter Account Title:"); /\*argument passs to constractor with email\*/

acTitle=s2.nextLine();

System.out.println("Enter opening Balance:(minimum 5000)");

obal=s.next();

openingBal=Double.parseDouble(obal) ;

if(openingBal<5000){

throw new IllegalArgumentException("Minimum Balance should be 5000\n Try again");

}

Account account1= new Account(i,acTitle,openingBal);

acc[i]=account1;

System.out.println(+i+" no Account created Successfully. \n");

break;

default:

i--;

System.out.println("You have pressed wrong button.");

System.out.println("Please press either the number #1 or the number #2 & confirm your answer");

} //switch end;

} //Try End

catch(NumberFormatException e)

{

System.out.println("Unable to format.Try again.Use only numerical value here");

i--;

}

catch(IllegalArgumentException e1)

{

System.out.println("Minimum Balance should be 5000.\nTry again");

i--;

}

catch(NullPointerException e3)

{

System.out.println("Null Value not accepted.\n Try again");

i--;

}

catch(ArithmeticException e4){

System.out.println("Arithmetic exception Found .\n Try again ");

i--;

}

catch(InputMismatchException e5){

System.out.println("Wrong Input .\n Try again ");

}

} //for loop end;

outer:

for(;;){

try{

System.out.println("Press Any button:");

System.out.println("1.Deposit");

System.out.println("2.Withdraw");

System.out.println("3.Set new Account Balance");

System.out.println("4.Set email id");

System.out.println("5.Print a single Account Information");

System.out.println("6.Print All Account Information");

System.out.println("7.Exit");

int p=s.nextInt();

switch(p){ //to control user commands

case 1:

System.out.println("Enter Account id:");

int idCheck=s.nextInt();

System.out.println("Enter Deposit Amount:");

double amount=s.nextDouble();

if(acc[idCheck].deposit(amount)==true){ //Deposit

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

}

else if(acc[idCheck].deposit(amount)==false){

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

}

break;

case 2:

System.out.println("Enter Account id:");

idCheck=s.nextInt();

System.out.println("Enter withdraw Amount below:");

amount=s.nextDouble();

if(acc[idCheck].Withdraw(amount)==true){

System.out.println("Withdraw Successful.");

}

else if(acc[idCheck].Withdraw(amount)==false){

System.out.println("withdraw Failed.Minimum Balance should remain 100 after withdrawal");

}

break;

case 3:

System.out.println("Enter Account id:");

idCheck=s.nextInt();

System.out.println("Enter New balance:");

amount=s.nextDouble();

if(acc[idCheck].setAccountBal(amount)==true){

System.out.println("Setting new Account Balance Successful.\n");

}

else if(acc[idCheck].setAccountBal(amount)==false){

System.out.println("Setting new Account Balance Failed.");

}

break;

case 4:

System.out.println("Enter Account id:");

idCheck=s.nextInt();

System.out.println("Enter Email address:");

email=s.next();

acc[idCheck].setEmailID(email);

System.out.println("Email set Successfully.");

break;

case 5: //account id's are automatically generated respectively

System.out.println("Enter Account id to print it's Information:");

idCheck=s.nextInt();

System.out.println(""+acc[idCheck].toString());

break;

case 6:

System.out.println("Here is All Account Information:");

for (int i = 1; i <=x; i++) {

System.out.println("..............."+acc[i].toString());

}

case 7:

break outer;

}

} //try end

catch(NumberFormatException e)

{

System.out.println("Unable to format.Try Again");

}

catch(IllegalArgumentException e1)

{

System.out.println("Minimum Balance should be 5000.\nTry again");

}

catch(NullPointerException e3)

{

System.out.println("Null Value not accepted.\n Try again");

}

catch(ArithmeticException e4){

System.out.println("Arithmatic exception Found .\n Try again ");

}

catch(InputMismatchException e5){

System.out.println("Wrong Input .\n Try again ");

}

} //for loop end;

}

}

**Main End**

package testaccounts;

import java.util.Objects;

public class Account {

private int Accountid;

private String accountTitle;

private double openingBal;

private String emailID;

private double accountBal=0;

private static int minBalLimit=100;

private static int numAccounts=0;

public Account(int Accountid, String accountTitle, double openingBal) {

this.Accountid = Accountid;

this.accountTitle = accountTitle;

this.openingBal = openingBal;

this.accountBal =openingBal;

numAccounts++;

}

public Account(int Accountid, String accountTitle, double openingBal, String emailID) {

this.Accountid = Accountid;

this.accountTitle = accountTitle;

this.openingBal = openingBal;

this.accountBal =openingBal;

this.emailID = emailID;

numAccounts++;

}

public void setAccountid(int Accountid) {

this.Accountid = Accountid;

}

public void setAccountTitle(String accountTitle) {

this.accountTitle = accountTitle;

}

public void setOpeningBal(double openingBal) {

this.openingBal = openingBal;

this.accountBal=openingBal;

}

public void setEmailID(String emailID) {

this.emailID = emailID;

}

public boolean setAccountBal(double newaccountBal) {

if(newaccountBal>=minBalLimit){

this.accountBal = newaccountBal;

return true;

}

else{

return false;

}

}

public int getAccountid() {

return Accountid;

}

public String getAccountTitle() {

return accountTitle;

}

public double getOpeningBal() {

return openingBal;

}

public String getEmailID() {

return emailID;

}

public double getAccountBal() {

return accountBal;

}

public static int getMinBalLimit() {

return minBalLimit;

}

public static int getNumAccounts() {

return numAccounts;

}

public boolean deposit(double amount){

accountBal+=amount;

return true;

}

public boolean Withdraw(double amount){

if(accountBal-amount>=minBalLimit){

accountBal-=amount;

return true;

}

else{

return false;

}

}

@Override

public String toString() {

return "........\n Account id=" + Accountid + ",\n Account Title: " + accountTitle + ",\n Opening Balance: " + openingBal + ",\n Email address: " + emailID + ", \n Account Balance: " + accountBal ;

}

@Override

public int hashCode() {

int hash = 5;

return hash;

}

public boolean equals(Account obj) {

if (this == obj) {

return true;

}

if (obj == null) {

return false;

}

if (getClass() != obj.getClass()) {

return false;

}

final Account other = (Account) obj;

if (this.Accountid != other.Accountid) {

return false;

}

if (Double.doubleToLongBits(this.openingBal) != Double.doubleToLongBits(other.openingBal)) {

return false;

}

if (Double.doubleToLongBits(this.accountBal) != Double.doubleToLongBits(other.accountBal)) {

return false;

}

if (!Objects.equals(this.accountTitle, other.accountTitle)) {

return false;

}

if (!Objects.equals(this.emailID, other.emailID)) {

return false;

}

return true;

}

}