**Experiment 1.3**

**Student Name:Shashi Ranjan Mehta UID: 21BCS7093**

**Branch:BE-CSE Section/Group:CC-FL-601**

**Semester: 6 Date of Performance:31-01-2024**

**Subject Name: Java Lab**

**Subject Code:21CSH-319**

1. **Aim:** Calculate interest based on the type of the account and the status of the account holder
2. **Objective:** The goal of this project is to design and implement Calculate interest based on the type of the account and the status of the account holder. The rates of interest changes according to the amount (greater than or less than 1 crore), age of account holder (General or Senior citizen) and number of days if the type of account is FD or RD.
3. **Algo. /Approach and output:**

import java.util.Scanner;

abstract class Account {

double interestRate;

double amount;

abstract double calculateInterest() throws IllegalArgumentException;

}

class FDAccount extends Account {

int noOfDays;

int ageOfAccountHolder;

@Override

double calculateInterest() throws IllegalArgumentException {

if (amount <= 0 || noOfDays <= 0) {

throw new IllegalArgumentException("Amount and number of days must be positive values.");

}

if (amount <= 10000000) {

if (noOfDays >= 7 && noOfDays <= 14)

return amount \* 0.045;

else if (noOfDays >= 15 && noOfDays <= 29)

return amount \* 0.0475;

else if (noOfDays >= 30 && noOfDays <= 45)

return amount \* 0.055;

} else {

if (noOfDays >= 7 && noOfDays <= 14)

return amount \* 0.065;

else if (noOfDays >= 15 && noOfDays <= 29)

return amount \* 0.0675;

}

throw new IllegalArgumentException("Invalid number of days.");

}

}

class SBAccount extends Account {

@Override

double calculateInterest() throws IllegalArgumentException {

if (amount <= 0) {

throw new IllegalArgumentException("Amount must be a positive value.");

}

return amount \* 0.04;

}

}

class RDAccount extends Account {

int noOfMonths;

double monthlyAmount;

@Override

double calculateInterest() throws IllegalArgumentException {

if (monthlyAmount <= 0 || noOfMonths <= 0) {

throw new IllegalArgumentException("Monthly amount and number of months must be positive values.");

}

if (noOfMonths == 6)

return monthlyAmount \* 6 \* 0.075;

else if (noOfMonths == 9)

return monthlyAmount \* 9 \* 0.0775;

else if (noOfMonths == 12)

return monthlyAmount \* 12 \* 0.08;

throw new IllegalArgumentException("Invalid number of months.");

}

}

public class calculator{

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

while (true) {

System.out.println("Select the option:");

System.out.println("1. Interest Calculator –SB");

System.out.println("2. Interest Calculator –FD");

System.out.println("3. Interest Calculator –RD");

System.out.println("4. Exit");

int option = scanner.nextInt();

switch (option) {

case 1:

SBAccount sbAccount = new SBAccount();

System.out.println("Enter the Average amount in your account:");

sbAccount.amount = scanner.nextDouble();

try {

System.out.println("Interest gained: Rs. " + sbAccount.calculateInterest());

} catch (IllegalArgumentException e) {

System.out.println("Error: " + e.getMessage());

}

break;

case 2:

FDAccount fdAccount = new FDAccount();

System.out.println("Enter the FD amount:");

fdAccount.amount = scanner.nextDouble();

System.out.println("Enter the number of days:");

fdAccount.noOfDays = scanner.nextInt();

System.out.println("Enter your age:");

fdAccount.ageOfAccountHolder = scanner.nextInt();

try {

System.out.println("Interest gained is: " + fdAccount.calculateInterest());

} catch (IllegalArgumentException e) {

System.out.println("Error: " + e.getMessage());

}

break;

case 3:

RDAccount rdAccount = new RDAccount();

System.out.println("Enter the RD monthly amount:");

rdAccount.monthlyAmount = scanner.nextDouble();

System.out.println("Enter the number of months:");

rdAccount.noOfMonths = scanner.nextInt();

try {

System.out.println("Interest gained is: " + rdAccount.calculateInterest());

} catch (IllegalArgumentException e) {

System.out.println("Error: " + e.getMessage());

}

break;

case 4:

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

System.exit(0);

break;

default:

System.out.println("Invalid option. Please select again.");

break;

}

}

}

}

Output:



