Name of the Course : Complete Java SE8 Developer Bootcamp

Level : Difficult

Tool Stack : Java8 and Junit5

Problem Statement : A Banking application like to give discount for account holders based on

the transaction amount and registration date .

The user will provide account details , that would include id (string), dor(date of registration)(string) and transaction amount (integer).

Provide two hash maps , the first one has employee id as key and for as value.The second hashmap contains amount as value and id as key.

The business rule is

1. for transaction amount greater than n or equal to 20000 and for >= 5 year registration , the discount rate is 20% of TA.
2. For Transaction Amount >= 20000 and for less than 5 years registration the discount rate is 10%.
3. For TA < 20000 and registration greater than or equal to 5 year, the discount rate is 15 %
4. if TA less than 20000 and for registration less than 5 year, the discount rate is 5%.

Description : Create class BankRate with a method calculateDiscount() which takes the two hash maps as input and returns the treemap as output .

provide a main method to get account holder details.The dor(date of registration)format is

“dd-mm-yyyy”

Output will be user id and discount amount in one line for each user.

Sample Input :

4

A-1010  
20-11-2007  
25000  
B-1011  
04-12-2010  
30000  
C-1012  
11-11-2005  
15000  
D-1013  
02-12-2012  
10000  
**Sample Output 1:**  
A-1010:5000  
B-1011:3000  
C-1012:2250  
D-1013:500

Code:

**import java.text.ParseException;**

**import java.text.SimpleDateFormat;**

**import java.time.LocalDate;**

**import java.time.Period;**

**import java.time.format.DateTimeFormatter;**

**import java.time.format.ResolverStyle;**

**import java.util.Date;**

**import java.util.HashMap;**

**import java.util.Iterator;**

**import java.util.Scanner;**

**import java.util.TreeMap;**

**public class BankRate {**

**public static void main(String []args) throws ParseException{**

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

**int s=Integer.parseInt(sc.nextLine());**

**HashMap<String,String>hm=new HashMap<String,String>();**

**HashMap<String,Integer>hm1=new HashMap<String,Integer>();**

**for(int i=0;i<s;i++)**

**{**

**String id=sc.nextLine();**

**hm.put(id, sc.nextLine());**

**hm1.put(id,Integer.parseInt(sc.nextLine()));**

**}**

**TreeMap<String,Integer>tm=new TreeMap<String,Integer>();**

**tm = calculateDiscount(hm,hm1);**

**Iterator<String> it=tm.keySet().iterator();**

**while(it.hasNext())**

**{**

**String n=it.next();**

**int fac=tm.get(n);**

**System.out.println(n+":"+fac);**

**}**

**}**

**public static TreeMap<String,Integer> calculateDiscount(HashMap<String,String>hm,HashMap<String,Integer>hm1) throws ParseException**

**{**

**TreeMap<String,Integer> tm=new TreeMap<String,Integer>();**

**// SimpleDateFormat sdf=new SimpleDateFormat("dd-MM-yyyy");**

**DateTimeFormatter df = DateTimeFormatter.ofPattern("dd-MM-uuuu")**

**.withResolverStyle(ResolverStyle.STRICT);**

**Iterator<String> itr1=hm.keySet().iterator();**

**while(itr1.hasNext())**

**{**

**try**

**{**

**String id=itr1.next();**

**String dor=hm.get(id);**

**int am=hm1.get(id);**

**LocalDate d1=LocalDate.parse(dor, df);**

**String s1="01-01-2015";**

**LocalDate d2= LocalDate.parse(s1,df);**

**int exp=Period.between(d1, d2).getYears();**

**if(am>=20000 && exp>=5)**

**{**

**int dis=(int) (0.20 \* am);**

**tm.put(id,dis);**

**}**

**else if(am>=20000 && exp<5)**

**{**

**int dis=(int) (0.1\*am);**

**tm.put(id,dis);**

**}**

**else if(am<20000 && exp>=5)**

**{**

**int dis=(int) (0.15\*am);**

**tm.put(id,dis);**

**}**

**else if(am<20000 && exp<5)**

**{**

**int dis=(int) (0.05\*am);**

**tm.put(id,dis);**

**}**

**}**

**catch(Exception e){**

**System.out.println(e);**

**}**

**}**

**return tm;**

**}**

**}**

Junit Testing

**import** java.text.ParseException;

**import** java.util.HashMap;

**import** java.util.TreeMap;

**import** org.junit.Assert;

**import** org.junit.Test;

**import** handson.BankRate;

**import** handson.DateValidator;

**public** **class** TestBankRate {

@Test

**public** **void** testBankRate\_Success() **throws** ParseException {

HashMap<String,String> hm = **new** HashMap<String,String>();

hm.put("A-1010", "20-11-2007");

hm.put("B-1011","04-12-2010");

hm.put("C-1012", "11-11-2005");

hm.put("D-1013","02-12-2012");

HashMap<String,Integer> hm1= **new** HashMap<String,Integer>();

hm1.put("A-1010", 25000);

hm1.put("B-1011",30000);

hm1.put("C-1012", 15000);

hm1.put("D-1013",10000);

TreeMap<String,Integer> tm=**new** TreeMap<String,Integer>();

tm.put("A-1010",5000);

tm.put("B-1011",3000);

tm.put("C-1012",2250);

tm.put("D-1013",500);

Assert.*assertEquals*(tm, BankRate.*calculateDiscount*(hm,hm1));

}

}

Test Data1

**Sample Input 1:**  
4  
A-1010  
20-11-2007  
25000  
B-1011  
04-12-2010  
30000  
C-1012  
11-11-2005  
15000  
D-1013  
02-12-2012  
10000  
**Sample Output 1:**  
A-1010:5000  
B-1011:3000  
C-1012:2250  
D-1013:500

Learning outcome: Participant could able to learn how to use the Collections API .