COMP1406 - Assignment #9

(Due: **Monday, March 24th @ 11:30pm)**

In this assignment, you will get practice using **ArrayList** and **HashMap** data types.

(1) The Basic Classes

Assume that a private cell phone company wanted you to write a program to keep track of various calls made between its customers. Here is a basic **Customer** class that maintains information for a single customer:

```
public class Customer {
   private String       name;
   private String       number;
   private PhonePlan    plan;

   public Customer(String n, String num, PhonePlan p) {
      name = n;
      number = num;
      plan = p;
   }

   public String    getName() { return name; }
   public String    getNumber() { return number; }
   public PhonePlan getPlan() { return plan; }
   public void      setPlan(PhonePlan p) { plan = p; }

   public String toString() {
      return name + " [" + number + "] on a " + plan.toString();
   }
}
```

The above class makes use of a **PhonePlan** class as follows:

```
public class PhonePlan {
   private int        minutesAllowed;
   private int        minutesUsed;

   public PhonePlan(int mins) {
      minutesAllowed = mins;
      minutesUsed = 0;
   }

   public int getMinutesAllowed() { return minutesAllowed; }
   public int getMinutesUsed() { return minutesUsed; }
   public void setMinutesAllowed(int x) { minutesAllowed = x; }
   public void setMinutesUsed(int x) { minutesUsed = x; }

   public int getMinutesRemaining() {
      return minutesAllowed - minutesUsed;
   }

   public String toString() {
      return ("(" + minutesAllowed + " minute) Monthly Plan with " +
               getMinutesRemaining() + " minutes remaining");
   }
}
```

You will need this **Call** class as well, which represents information corresponding to a phone call:

```
import java.util.Date;

public class Call {
      private Customer  madeFrom;
      private Customer  madeTo;
      private Date      time;
      private int       length;     // in seconds

      public Call(Customer f, Customer t, Date d, int len) {
            madeFrom = f;
            madeTo = t;
            time = d;
            length = len;
      }

      public Customer getMadeFrom() { return madeFrom; }
      public Customer getMadeTo() { return madeTo; }
      public Date getTime() { return time; }
      public int getLength() { return length; }

      public String toString() {
            return (String.format("%d:%02d",length/60,length%60) +
                    " call from " + madeFrom.getNumber() +
                    " to " + madeTo.getNumber());
      }
}
```

(2) The **PhoneNetwork** class

Create a class called **PhoneNetwork** that keeps an **ArrayList** called customers of **Customer** objects to represent the customers that will be making and receiving phone calls. The class must also keep two **HashMaps** called outgoingCalls and incomingCalls. The keys for the outgoing call map are the unique phone numbers of customers who have *made* calls during the month. The value for each key of the map should be an **ArrayList** of **Call** objects that represents all calls that were *made* from that phone number that month (i.e., could be empty). Similarly the keys for the incoming call map are the unique phone numbers of customers who have *received* calls during the month. The value for each key of the map should also be an **ArrayList** of **Call** objects that represents all calls that were *received* to that phone number that month (i.e., could be empty).

1. Create an appropriate constructor and get methods.

2. Create a **register**() method that takes a customer's name (String), phone number (String) and an integer indicating how many minutes that customer's monthly plan will allow before being charged extra. The code must maintain that information accordingly in the **PhoneNetwork** object.

3. Create a **makeCall**() method that takes a fromPhoneNumber, a toPhoneNumber and an integer indicating how many seconds that the call lasted. The code must update the **PhoneNetwork** object's HashMaps accordingly. Note that **new Date()** returns the current date and time. Note that for a 300 second call, this uses up 5 minutes of BOTH the customer calling as well as the customer being called. a 301 second call will use up 6 minutes (rounded up to nearest minute).

4. Write a **displayStats**() method that displays (to the **System.out** console) some statistics for each customer pertaining to the phone call activity during the month. It should show the customer's phone number, name, number of outgoing calls made (possibly 0), number of incoming calls received (possibly 0), the number of minutes per month allowed on their plan, the number of

minutes used during the month, the number of minutes gone over the plan limit, the base cost for the plan (which is $20 for the first 100 minutes plus $5 for every additional 100 minutes), the extra cost (which is 15 cents per minute over the plan's allowed minutes amount ... round seconds up to the nearest minute), the HST on the base amount plus extra cost, and finally the total cost. You should lay everything out neatly using **String.format()**. Here is what it should look like:

```
PHONE NUMBER    NAME                 OUT  IN     PLAN     USED     OVER      BASE     EXTRA       HST     TOTAL
613-111-1111    Rob Banks              7   2      200       74        0     25.00      0.00      3.25    $28.25
819-222-2222    April Rain             6   5      200       75        0     25.00      0.00      3.25    $28.25
613-333-3333    Rita Book              6   5      100       76        0     20.00      0.00      2.60    $22.60
613-444-4444    Sue Permann            8   7      100      116       16     20.00      2.40      2.91    $25.31
819-555-5555    Tim Bur                1   7      100       66        0     20.00      0.00      2.60    $22.60
613-666-6666    Paddy O'Lantern        3   5      100       74        0     20.00      0.00      2.60    $22.60
613-777-7777    Sam Pull               7   7      200      127        0     25.00      0.00      3.25    $28.25
613-888-8888    Sandy Beach            5   3      300       88        0     30.00      0.00      3.90    $33.90
819-999-9999    Adam Bomm              3   2      300       46        0     30.00      0.00      3.90    $33.90
613-555-1234    Hugo First             5   5      300       75        0     30.00      0.00      3.90    $33.90
613-555-5678    Lee Nover              5   0      200       48        0     25.00      0.00      3.25    $28.25
613-666-1234    Mabel Syrup            3   5      200       51        0     25.00      0.00      3.25    $28.25
613-666-5678    Mike Rohsopht          6   5      300      114        0     30.00      0.00      3.90    $33.90
613-777-1234    Adam Sapple            2   1      100       26        0     20.00      0.00      2.60    $22.60
613-777-5678    Moe Skeeto             3   8      100       62        0     20.00      0.00      2.60    $22.60
819-888-1234    Anita Bath             4   5      100       78        0     20.00      0.00      2.60    $22.60
613-888-5678    Rusty Chain            6   3      100       96        0     20.00      0.00      2.60    $22.60
613-999-1234    Stu Pitt               6   8      200       97        0     25.00      0.00      3.25    $28.25
613-999-5678    Val Crow               7   5      300      100        0     30.00      0.00      3.90    $33.90
613-444-1234    Neil Down              2   7      300       85        0     30.00      0.00      3.90    $33.90
```

5. Create a method called **customerMakingMostCalls()** which returns a **Customer** object representing the customer who made the most calls during the month.

6. Create a method called **customerReceivingMostCalls()** which returns a **Customer** object representing the customer who received the most calls during the month.

7. Create a method called **wasCallMade(String n1, String n2)** which returns a **boolean** indicating whether or not a call was made during the month from phone number n1 to phone number n2.

Here is the program that you must run to test your code. Since it generates up to 100 random calls, make sure to run it a few times. Check to make sure that it works for all these situations:

- at least one Customer did not make any calls
- at least one Customer did not receive any calls
- a situation where "Stu Pitt" called "Mabel Syrup"
- a situation where "Stu Pitt" did not call "Mabel Syrup".

```java
public class CallSimulationTestProgram {
   public static void main(String args[]) {

         PhoneNetwork   phoneNetwork = new PhoneNetwork();

         // Register some customers
         phoneNetwork.register("Rob Banks", "613-111-1111", 200);
         phoneNetwork.register("April Rain", "819-222-2222", 200);
         phoneNetwork.register("Rita Book", "613-333-3333", 100);
         phoneNetwork.register("Sue Permann", "613-444-4444", 100);
         phoneNetwork.register("Tim Bur", "819-555-5555", 100);
         phoneNetwork.register("Paddy O'Lantern", "613-666-6666", 100);
         phoneNetwork.register("Sam Pull", "613-777-7777", 200);
         phoneNetwork.register("Sandy Beach", "613-888-8888", 300);
         phoneNetwork.register("Adam Bomm", "819-999-9999", 300);
         phoneNetwork.register("Hugo First", "613-555-1234", 300);
         phoneNetwork.register("Lee Nover", "613-555-5678", 200);
         phoneNetwork.register("Mabel Syrup", "613-666-1234", 200);
         phoneNetwork.register("Mike Rohsopht", "613-666-5678", 300);
         phoneNetwork.register("Adam Sapple", "613-777-1234", 100);
```

```
        phoneNetwork.register("Moe Skeeto", "613-777-5678", 100);
        phoneNetwork.register("Anita Bath", "819-888-1234", 100);
        phoneNetwork.register("Rusty Chain", "613-888-5678", 100);
        phoneNetwork.register("Stu Pitt", "613-999-1234", 200);
        phoneNetwork.register("Val Crow", "613-999-5678", 300);
        phoneNetwork.register("Neil Down", "613-444-1234", 300);

        // Simulate up to 100 calls randomly
        for (int i=0; i<100; i++) {
           int ind = (int)(Math.random()*phoneNetwork.getCustomers().size());
           Customer fromPerson = phoneNetwork.getCustomers().get(ind);
           ind = (int)(Math.random()*phoneNetwork.getCustomers().size());
           Customer toPerson = phoneNetwork.getCustomers().get(ind);

           int callLength = (int)(Math.random()*1000 + 5);
           if (fromPerson != toPerson)
              phoneNetwork.makeCall(fromPerson.getNumber(),
                           toPerson.getNumber(), callLength);
        }

        // Add a 5 minute call from Rusty Chain to April Rain
        phoneNetwork.makeCall("613-888-5678", "819-222-2222", 300);

        // Display the stats
        phoneNetwork.displayStats();

        // Compute some interesting details
        System.out.println("\nCustomer who made the most calls: " +
                 phoneNetwork.customerMakingMostCalls());
        System.out.println("Customer who received the most calls: " +
                 phoneNetwork.customerReceivingMostCalls());
        System.out.println("Did Rusty Chain call April Rain: " +
                 phoneNetwork.wasCallMade("613-888-5678", "819-222-2222"));
        System.out.println("Did Rusty Chain call himself: " +
                 phoneNetwork.wasCallMade("613-888-5678", "613-888-5678"));
        System.out.println("Did Stu Pitt call Mabel Syrup: " +
                 phoneNetwork.wasCallMade("613-999-1234", "613-666-1234"));
      }
   }
```

Make sure that your test code compiles, runs and produces the correct output. Hand this test code in with your assignment.

NOTE: Submit all **.java** files needed to run (including the test program). You **MUST NOT use packages** in your code, **nor projects**. Submit ALL of your files in one folder such that they can be opened and compiled individually in JCreator. Some IDEs may create packages and/or projects automatically. You **MUST** export the .java files and remove the package code at the top if it is there. Do NOT submit JCreator projects either. **JUST SUBMIT the JAVA FILES**. Note that if your internet connection at home is down or does not work, we will not accept this as a reason for handing in an assignment late ... so make sure to submit the assignment WELL BEFORE it is due !

Please NOTE that you WILL lose marks on this assignment if any of your files are missing. You will also lose marks if your code is not written neatly with proper indentation. See examples in the notes for proper style.
