

```

/*
    Problem 7A:      New methods for Number of Meters per site
    Programmer:     Wayne Spence
    Date:           November 10, 2003
    Program Name:   DME7ACustomer
*/

import javax.swing.*;
import java.math.*;

public class DME7ACustomer
{
    //Declare variables
    private String customerNameString;
    private String customerTypeString;
    private int customerNumber;
    private int serviceType;
    private int meterReadingDay;
    private int meterReadingMonth;
    private int meterReadingYear;
    private int lastReading;
    private int currentReading;
    private int number_of_meters_at_site;

    private String outputLine = "";
    public static final String DME_PANE_TITLE = "Denton Municipal Utilities";

    CustomerFind customerFind = new CustomerFind();

public DME7ACustomer()
{
    customerNameString = "";
    customerTypeString = "R";
    customerNumber= 0;
    serviceType= 1;
    meterReadingDay = 5;
    meterReadingMonth = 10;
    meterReadingYear = 2003;
    currentReading = 0;
    number_of_meters_at_site = 0;
}
}

```

```

public void setCustomerNumber()
{
    boolean valid = false;
    while (!valid)
    {
        customerNumber = promptUser("customer number", customerNumber, 1, 99999);
        valid = customerFind.getCustomerNumberDB(customerNumber);
    }
}

public void setCustomerType()
{
    customerTypeString = customerFind.getCustomerTypeDB();
    customerTypeString = promptUser ("customer type (R, C or G)", customerTypeString);
}

public void setServiceType()
{
    serviceType = customerFind.getServiceTypeDB();
    serviceType = promptUser ("service type (1 or 2)", serviceType, 1, 2);
}

public void setMeterReadingDay(int currentDayValue)
{
    meterReadingDay = customerFind.getMeterReadingDayDB();
    meterReadingDay = promptUser ("meter reading day", meterReadingDay, currentDayValue, 1, 31);
}

public void setMeterReadingMonth(int currentMonthValue)
{
    meterReadingMonth = customerFind.getMeterReadingMonthDB();
    meterReadingMonth = promptUser ("meter reading month", meterReadingMonth, currentMonthValue, 1,
12);
}

public void setMeterReadingYear(int currentYearValue)
{
    int lowestYearValue = currentYearValue - 3;
    meterReadingYear = customerFind.getMeterReadingYearDB();
    meterReadingYear = promptUser ("meter reading year", meterReadingYear, currentYearValue,
lowestYearValue, currentYearValue);
}

```

```
public void setCurrentReading()
{
    lastReading = customerFind.getLastReadingDB();
//    mock-up to represent different values for old meter reading
    lastReading = (int)(Math.random() * 100000.0);
    currentReading = promptUser ("kilowatt hours usage (KWH)", lastReading, 0, 99999);
}

public void setNumberOfMeters(int defaultMeters)
{
    number_of_meters_at_site =
        promptUser ("number of meters at site", defaultMeters, 1, 10);
}

public int getCustomerNumber()
{
    return customerNumber;
}

public String getCustomerName()
{
    customerNameString = customerFind.getCustomerNameDB();
    return customerNameString;
}

public String getCustomerType()
{
    return customerTypeString;
}

public int getServiceType()
{
    return serviceType;
}

public int getMeterReadingDay()
{
    return meterReadingDay;
}

public int getMeterReadingMonth()
```

```
{
    return meterReadingMonth;
}

public int getMeterReadingYear()
{
    return meterReadingYear;
}

public int getLastReading()
{
    return lastReading;
}

public int getCurrentReading()
{
    return currentReading;
}

public int getNumberofMeters()
{
    return number_of_meters_at_site;
}

private static int promptUser (String userPrompt, int initialValue, int minValue, int maxValue)
{
    boolean goodData = false;
    String outputLine = "";
    String initialValueString = Integer.toString(initialValue);
    int intNumber = 0;

    while (!goodData)
    {
        String intNumberString =
            JOptionPane.showInputDialog(null, "What is the " + userPrompt + "?", initialValueString);
        try
        {
            intNumber = Integer.parseInt(intNumberString);
            if (intNumber < minValue || intNumber > maxValue)
            {
                outputLine = "The " + userPrompt + " should be between " +
                    minValue + " and " + maxValue + ".";
                displayMessage(outputLine);
            }
        }
    }
}
```

```

        }
    else
        goodData = true;
    }
    catch (NumberFormatException e)
    {
        outputLine = "You have not entered digits for the " + userPrompt + ".";
        displayMessage(outputLine);
    }
}
return intNumber;
}

private static int promptUser (String userPrompt, int lastValue, int initialValue, int minValue, int
maxValue)
{
    boolean goodData = false;
    String outputLine = "";
    String initialValueString = Integer.toString(initialValue);
    int intNumber = 0;
    String userPromptExtended = "What is the " + userPrompt +
        "? The last reading indicated " + lastValue + ".";

    while (!goodData)
    {
        String intNumberString =
            JOptionPane.showInputDialog(null, userPromptExtended , initialValueString);
        try
        {
            intNumber = Integer.parseInt(intNumberString);
            if (intNumber < minValue || intNumber > maxValue)
            {
                outputLine = "The " + userPrompt + " should be between " +
                    minValue + " and " + maxValue + ".";
                displayMessage(outputLine);
            }
        }
        else
            goodData = true;
    }
    catch (NumberFormatException e)
    {
        outputLine = "You have not entered digits for the " + userPrompt + ".";
        displayMessage(outputLine);
    }
}

```

```
        }
    }
    return intNumber;
}

private static String promptUser (String userPrompt, String initialValue)
{
    boolean goodData = false;
    String outputLine = "";
    String stringData = "";
    while (!goodData)
    {
        stringData =
            JOptionPane.showInputDialog(null, "What is the " + userPrompt + "?",
                initialValue);
        goodData = true;
    }
    return stringData;
}

private static void displayMessage (String messageText)
{
    String errorTitle = "An error has occurred";
    messageText = messageText + "\nPlease, try again!";

    JOptionPane.showMessageDialog(null, messageText, errorTitle,
        JOptionPane.ERROR_MESSAGE);
}
}
```