

```

/*
    Problem 5 :      DME Customer class for creating
                    customer objects
    Programmer:      Wayne Spence
    Date:            September 16, 2003
    Program Name:    DME4DCustomer
*/
import javax.swing.*;
import java.sql.*;

public class DME5BCustomer
{
    //Declare instance variables
    private String customerNameString;
    private String customerTypeString;
    private String customerNumberString;
    private int serviceType;
    private int meterReadingDay;
    private int meterReadingMonth;
    private int meterReadingYear;
    private int lastReading;
    private int currentReading;

    CustomerFindDB customerDB = new CustomerFindDB();
    Customer customer = new Customer();

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

    public DME5BCustomer()
    {
        customerNameString = "";
        customerTypeString = "R";
        customerNumberString= "0";
        serviceType= 1;
        meterReadingDay = 9;
        meterReadingMonth = 2;
        meterReadingYear = 2004;
        lastReading = 0;
        currentReading = 0;
    }

    public boolean setCustomerNumber(String inputNumber)
    {
        boolean valid = false;
        boolean promptOK = false;
        customerNumberString = inputNumber;
        promptOK = promptUser("customer number", customerNumberString, 1,
99999);
        if (promptOK)
        {
            int customerNumber = Integer.parseInt(customerNumberString);
            try
            {
                customerDB.connect();

```

```

        customerDB.open();
        customer = customerDB.selectRecord(customerNumber);
        if (customer != null)
        {
            customerNameString = customer.getLastName() + ", " +
                customer.getFirstName() + " " +
customer.getMiddleName();
            customerTypeString = customer.getCustomerType();
            serviceType = customer.getServiceType();
            meterReadingMonth = customer.getLastMonth();
            meterReadingDay = customer.getLastDay();
            meterReadingYear = customer.getLastYear();
            lastReading = customer.getLastReading();
            valid = true;
        }
        else
        {
            throw new SQLException();
        }
        catch (ClassNotFoundException exite)
        {
            JOptionPane.showMessageDialog(null, "Could not Find Class"+
exite.getMessage());
            System.exit(1);
        }
        catch (SQLException sqle)
        {
            outputLine = "Customer number " + customerNumberString +
                " was not found in the database.\n"
                + "Please re-check your numbers and try again.";
            JOptionPane.showMessageDialog(null, outputLine);
        }
    }
    return valid;
}

public boolean setMeterReadingDay(String endMeterReading)
{
    boolean valid = false;
    valid = promptUser ("meter reading day", endMeterReading, 1, 31);
    return valid;
}

public void setMeterReadingMonth()
{
    meterReadingMonth = promptUser ("meter reading month",
meterReadingMonth, 1, 12);
}

public void setMeterReadingYear()
{
    meterReadingYear = promptUser ("meter reading year", meterReadingYear,
2000, 2003);
}

```

```

public void setCurrentReading()
{
    currentReading = promptUser ("kilowatt hours usage (KWH)--last
reading below", lastReading, 1, 99999);
}

public boolean setEndMeterReading(String endMeterReading)
{
    boolean valid = false;
    valid = promptUser ("last meter reading", endMeterReading, 1,
99999);
    return valid;
}

public String getCustomerNumber()
{
    return customerNumberString;
}

public String getCustomerName()
{
    customerNameString = customer.getLastName() + ", " +
customer.getFirstName() + " " + customer.getMiddleName();
    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;
    }

    private static boolean promptUser (String userPrompt, String
initialValueString, int minValue, int maxValue)
    {
        String outputLine = "";
        boolean valid = false;
        String intNumberString = "";
        int intNumber = 0;

        intNumberString = initialValueString;
        try
        {
            intNumber = Integer.parseInt(intNumberString);
            if (intNumber < minValue || intNumber > maxValue)
                {
                    outputLine = "The " + userPrompt + " should be
between " +
                    minValue + " and " + maxValue + ".";
                    displayMessage(outputLine);
                }
            else
                valid = true;
        }
        catch (NumberFormatException e)
        {
            outputLine = "You have not entered digits for the " +
userPrompt + ".";
            displayMessage(outputLine);
        }
        return valid;
    }

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

        while (!goodData)
        {
            intNumberString =
                JOptionPane.showInputDialog(null, "What is the " + userPrompt
+ "?", intNumberString);
            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);
}
}

```