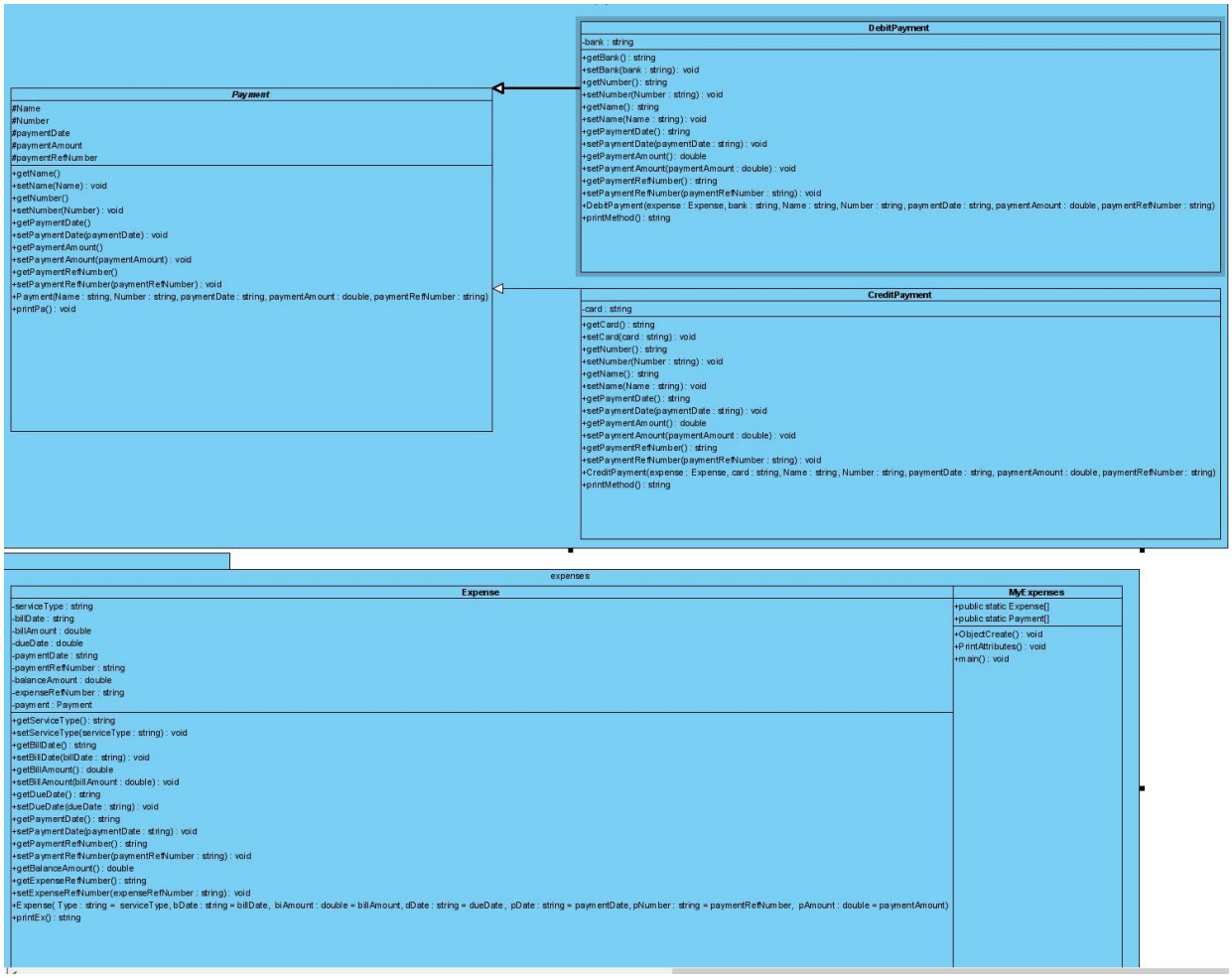
Ethan Couch
U79532295
IDE - IntelliJ

UML Class Diagram:



Source Code:

```
package expenses;
//Class MyExpenses in package expenses that contains:


import payments.Payment;
import payments.CreditPayment;
import payments.DebitPayment;
import java.io.*;
import java.util.Scanner;

public class MyExpenses {



    int numberLines = 0;
```

Ethan Couch
U79532295
IDE - IntelliJ

```
    int numberLines2 = 0;
   static Expense expense;
    static Payment payment;
    static String name = "Ethan Couch";

    //An array called myExpenses where all expense objects are stored.
    static int numberExpenses = 5;
    static Expense[] myExpenses = new Expense[numberExpenses];

    // An array called myPayments where all payment objects are stored.
    static int numberPayments = 5;
    static Payment[] myPayments = new Payment[numberPayments];


    //A "read" method where two files, "expenses.txt" and "payments.txt", will
read expense and payment data into the program. First read "expenses.txt" and
then "payments.txt". Note that the specific payment objects will only be
instantiated depending on the specific type of payment, debit or credit card.
Payments will use the expenseRefNumber to find the corresponding payments to
expense. You cannot enter your specific data into the program constructors.
    static void read()  {

        try (BufferedReader br = new BufferedReader(new
FileReader("expenses.txt"))) {
            String line;

            if ((line = br.readLine()) != null)
                numberExpenses = Integer.parseInt(line);
            else {
                System.out.println("Incorrect file format");
                return;
            }
//reads out expenses
            int counter = 0;
            while ((line = br.readLine()) != null) {
                String[] info = line.split(",");
                        Expense myExpArray = new Expense(info[0], info[1],
info[2], Double.parseDouble(info[3]), info[4]);
                        myExpenses[counter] = myExpArray;
                        counter++;
            }

        } catch (FileNotFoundException e) {
            e.printStackTrace();
        } catch (IOException e) {
            e.printStackTrace();
        }
```

Ethan Couch
U79532295
IDE - IntelliJ

```
        try (BufferedReader br2 = new BufferedReader(new
FileReader("payments.txt"))) {

/*
        if ((line2 = br2.readLine()) != null) {
            c = Integer.parseInt(line2);
            System.out.println(line2);
        }else {
            System.out.println("Incorrect file format");
            return;
        } */
//reads out payments
            int counter2 = 0;
            int c = Integer.parseInt(br2.readLine());
            for (int i = 0; i < c; i++) {
                String line2 = br2.readLine();
                String[] infoP = line2.split(",");



                    if (infoP[1].equals("Suntrust") || infoP[1].equals("Bank of
America")) {
                        myPayments[i] = new DebitPayment(myExpenses[i],
infoP[2], infoP[5], infoP[3], infoP[4], infoP[6],
Double.parseDouble(infoP[7]));
                        myExpenses[i].setPayment(myPayments[i]);
                    } else {
                        myPayments[i] = new CreditPayment(myExpenses[i],
infoP[2], infoP[5], infoP[3], infoP[4], infoP[6],
Double.parseDouble(infoP[7]));
                        myExpenses[i].setPayment(myPayments[i]);
                    }
                    counter2++;


            }
        }
        catch (FileNotFoundException e) {
            e.printStackTrace();
        } catch (IOException e) {
            e.printStackTrace();
        }

    }
        // An "init" method where all expense and payment objects are created.
Each expense object needs to be stored in myExpenses array. Each payment
object needs to be stored in myPayments array. In the program, each payment
corresponds to a single expense, and vice versa.
```

Ethan Couch
U79532295
IDE - IntelliJ

```
        public static void ObjectCreate()  {
            //MyExpenses my = new MyExpenses();
            //my.read();
            read();
        }

        // A "print" method printing all the attributes for all expense and
payment objects, and the total amount of expenses and the total balance after
payments.
        public static void printMethod()  {

            double totalExpenses = 0, totalExpensesBalance= 0;
            System.out.println("\nExpenses for "+name);
            for (int i=0; i<myExpenses.length; i++)  {
                payment = myExpenses[i].getPayment();
                totalExpenses += myExpenses[i].getBillAmount();
                myExpenses[i].printMethod();
            }
            System.out.println("\nPayments for "+name);
            for (int j=0; j<myPayments.length; j++)  {
                payment = myExpenses[j].getPayment();
                totalExpensesBalance +=myExpenses[j].getBillAmount() -
myPayments[j].getPaymentAmount();
                payment.printMethod();
            }
            System.out.println("\nTotal Expenses: " + totalExpenses);
            System.out.println("Total Balance: " + totalExpensesBalance);
            /*double totalExpenses = 0, totalExpensesBalance = 0;

            System.out.println("\nExpenses for " + name);
            for (int i = 0; i < numberExpenses; i++)  {
                payment = myExpenses[i].getPayment();
                myExpenses[i].printMethod();
                totalExpenses += myExpenses[i].getBillAmount();
                totalExpensesBalance += myExpenses[i].getBalanceAmount();
            }
            System.out.println("\nPayments for " + name);
            for (int i = 0; i < numberPayments; i++)  {
                payment = myExpenses[i].getPayment();
                payment.printMethod();
            }
            System.out.println("\nTotal Expenses: " + totalExpenses);
            System.out.println("Total Balance: " + totalExpensesBalance);*/
        }

// An "interact" method where the user will interact with the program, using
the Scanner class from Java. The user will enter an expenseRefNumber and the
program will print the corresponding expense and payment data.
public static void interact()  {
```

Ethan Couch
U79532295
IDE - IntelliJ

```
    Scanner scnr = new Scanner(System.in);
    String info = scnr.nextLine();

    while (scnr.hasNextLine()) {
        if(info.equals("Exit")) {
            break;
        }

        for (int i = 0; i < myExpenses.length; i++) {
            if (info.equals(myExpenses[i].getExpenseRefNumber())) {
                myExpenses[i].printMethod();
                //System.out.println("Expense Reference Number: " +
myExpenses[i].getExpenseRefNumber());
                myPayments[i].printMethod();
            }
            System.out.println("Expense Reference Number: " +
myExpenses[i].getExpenseRefNumber());
        }
    }
}
// The "main" method that calls the init, print, and interact methods.
    public static void main(String[]args){
        //MyExpenses my = new MyExpenses();
        //my.ObjectCreate();
        ObjectCreate();
        printMethod();
        interact();
        //my.printMethod();
       // my.interact();
    }
}
```

```
package payments;

// Subclass of Payment specific for credit payments. Class CreditPayment in
package payments contains:


import expenses.Expense;

public class CreditPayment extends Payment{

    //A private String data field named card storing the credit card name.
private String card;


//Methods that set and return the values for each variable.
```

Ethan Couch
U79532295
IDE - IntelliJ

```
    public String getCard() {
        return card;
    }
    public void setCard(String card) {
        this.card = card;
    }
    public String getNumber() {
        return Number;
    }
    public void setNumber(String number) {
        this.Number = number;
    }
    public String getName() {
        return Name;
    }
    public void setName(String name) {
        this.Name = name;
    }
    public String getPaymentDate() {
        return paymentDate;
    }
    public void setPaymentDate(String paymentDate) {
        this.paymentDate = paymentDate;
    }
    public double getPaymentAmount() {
        return paymentAmount;
    }
    public void setPaymentAmount(double paymentAmount) {
        this.paymentAmount = paymentAmount;
    }
    public String getPaymentRefNumber() {
        return super.getPaymentRefNumber();
    }
    public void setPaymentRefNumber(String paymentRefNumber) {
        super.setPaymentRefNumber(paymentRefNumber);
    }
```

```
    // A constructor method receiving all the arguments required to create each
payment, including the expense object.
public CreditPayment(Expense expense, String card,String paymentRefNumber,
String Name,  String Number, String paymentDate, double paymentAmount) {
  super(expense, paymentRefNumber, Name, Number, paymentDate,  paymentAmount);
    this.card = card;
}
```

Ethan Couch
U79532295
IDE - IntelliJ

```
// A print method printing the values of all variables. You may add additional
printing methods as needed
    public void printMethod() {
        System.out.println("\nExpense Reference Number: " +
expense.getExpenseRefNumber());
        System.out.println("Payment Reference Number: " + paymentRefNumber);
        System.out.println("CreditCard: "+card);
        System.out.println("Card Name: " + Name);
        System.out.println("Card Number: " + Number);
        System.out.println("Payment Date: " + paymentDate);;
        System.out.println("Payment Amount: " + paymentAmount);
    }

}
package payments;

// Subclass of Payment specific for debit payments. Class DebitPayment in
package payments contains:


import expenses.Expense;

public class DebitPayment extends Payment {

    // A private String data field named bank storing the bank name.
private String bank;

// Methods that set and return the values for each variable.

    public String getBank() {
        return bank;
    }
    public void setBank(String bank) {
        this.bank = bank;
    }
    public String getNumber() {
        return Number;
    }
    public void setNumber(String number) {
        this.Number = number;
    }
    public String getName() {
        return Name;
    }
    public void setName(String name) {
        this.Name = name;
    }
    public String getPaymentDate() {
        return paymentDate;
```

Ethan Couch
U79532295
IDE - IntelliJ

```
    }
    public void setPaymentDate(String paymentDate) {
        this.paymentDate = paymentDate;
    }
    public double getPaymentAmount() {
        return paymentAmount;
    }
    public void setPaymentAmount(double paymentAmount) {
        this.paymentAmount = paymentAmount;
    }
    public void setPaymentRefNumber(String paymentRefNumber) {
        super.setPaymentRefNumber(paymentRefNumber);
    }
    public String getPaymentRefNumber() {
        return super.getPaymentRefNumber();
    }

    // A constructor method receiving all the arguments required to create each
payment, including the expense object.
public DebitPayment( Expense expense, String bank, String paymentRefNumber,
String Name, String Number, String paymentDate, double paymentAmount){
    super(expense, paymentRefNumber, Name, Number, paymentDate, paymentAmount);
    this.bank = bank;
}

// A print method printing the values of all variables. You may add additional
printing methods as needed.
public void printMethod() {
    System.out.println("\nExpense Reference Number: " +
expense.getExpenseRefNumber());
    System.out.println("Payment Reference Number: " + paymentRefNumber);
    System.out.println("BankName: "+bank);
    System.out.println("Card Name: " + Name);
    System.out.println("Card Number: " + Number);
    System.out.println("Payment Date: " + paymentDate);;
    System.out.println("Payment Amount: " + paymentAmount);
}
}


package expenses;

import payments.*;

//Class Expense in package expenses. It implements the Print interface and
contains:
public class Expense {
    //public boolean getExpenseRefNumber;
    // A private String data field serviceType, e.g. "Electricity", "Gas", etc.
```

Ethan Couch
U79532295
IDE - IntelliJ

```
private String serviceType;
// A private String data field billDate, e.g. "Jan 4 2021".
    private String billDate;
//  A private double(or Double) data field billAmount, e.g. 25.00
    private double billAmount;
// A private String data field dueDate, e.g."Jan 30 2021".
    private String dueDate;

    private String paymentDate;

    private String paymentRefNumber;

    private double paymentAmount;
// A private double(or Double) data field named balanceAmount, computing the
difference between the billAmount and the paymentAmount for the expense.
    private double balanceAmount;

// A private String data field expenseRefNumber, e.g. "12345", etc.
    private String expenseRefNumber;


// A private Payment data field named payment,storing a reference to the
payment object where the corresponding payment information is stored.
    private Payment payment;


    //Methods that set and return the values for each variable. Note that for
balanceAmount there should only be a "get" method, and not a "set" method.

    public String getServiceType() {
        return serviceType;
    }

    public void setServiceType(String serviceType) {
        this.serviceType = serviceType;
    }

    public String getBillDate() {
        return billDate;
    }

    public void setBillDate(String billDate) {
        this.billDate = billDate;
    }

    public double getBillAmount() {
        return billAmount;
    }
```

Ethan Couch
U79532295
IDE - IntelliJ

```
    public void setBillAmount(double billAmount) {
        this.billAmount = billAmount;
    }

    public String getDueDate() {
        return dueDate;
    }
    public void setDueDate(String dueDate) {
        this.dueDate = dueDate;
    }

    public String getPaymentDate() {
        return paymentDate;
    }


    public void setPaymentDate(String paymentDate) {
        this.paymentDate = paymentDate;
    }


    public String getPaymentRefNumber() {
        return paymentRefNumber;
    }

    public void setPaymentRefNumber(String paymentRefNumber) {
        this.paymentRefNumber = paymentRefNumber;
    }

    public String getExpenseRefNumber() {
        return expenseRefNumber;
    }
    public double getBalanceAmount() {return billAmount - paymentAmount;}

    public void setExpenseRefNumber(String expenseRefNumber) {
        this.expenseRefNumber = expenseRefNumber;
    }

    public Payment getPayment() {
        return payment;
    }

    public void setPayment(Payment s) {
        payment = s;
        balanceAmount = billAmount - payment.getPaymentAmount();
    }

    //A constructor method receiving all the arguments required to create each
expense, except balanceAmount and payment.
```

Ethan Couch
U79532295
IDE - IntelliJ

```
    public Expense(String eNumber, String Type, String bDate, double biAmount,
String dDate) {
        setServiceType(Type);
        setBillDate(bDate);
        setBillAmount(biAmount);
        setDueDate(dDate);
        setExpenseRefNumber(eNumber);
        // paymentDate = pDate;
        //paymentRefNumber = pNumber;
       // paymentAmount = pAmount;
        //balanceAmount = billAmount - paymentAmount;
    }
//  A print method printing the values of all variables in the class.
    public void printMethod() {
        System.out.println("\nExpense Reference Number: " +expenseRefNumber);
        System.out.println("ServiceType: " + serviceType);
        System.out.println("BillDate: " + billDate);
        System.out.println("BillAmount: " + billAmount);
        System.out.println("DueDate: " + dueDate);
        System.out.println("BalanceAmount: " + balanceAmount);
        //System.out.println();
    }
}
package payments;
// Abstract class Payment in package payments. Superclass storing common
payment information. It implements the Print interface and contains:


import expenses.*;

public abstract class Payment implements Print {

    // A protected String data field named Name storing the name of the payment
account.
protected String Name;

// A protected String data field named Number storing the number of the
payment account.
protected String Number;

// A protected String data field paymentDate, e.g."Jan 20 2021".
protected String paymentDate;

// A protected double(or Double) data field named paymentAmount,e.g. 10.00
protected double paymentAmount;
```

Ethan Couch
U79532295
IDE - IntelliJ

```
// A protected String data field paymentRefNumber, e.g. "123456", etc. (Do not use your real reference information. The number may be composed of year, month, day, and any additional number, e.g. "20210101-9")
protected String paymentRefNumber;


// A protected Expense data field named expense, storing a reference to the expense object where the corresponding expense information is stored.
protected Expense expense;

// Methods that set and return the values for each variable.

    public String getName() {
        return Name;
    }
    public void setName(String name) {
        Name = name;
    }
    public String getNumber() {
        return Number;
    }
    public void setNumber(String number) {
        Number = number;
    }
    public String getPaymentDate() {
        return paymentDate;
    }
    public void setPaymentDate(String paymentDate) {
        this.paymentDate = paymentDate;
    }
    public double getPaymentAmount() {
        return paymentAmount;
    }
    public void setPaymentAmount(double s) {
        paymentAmount = s;
    }
    public String getPaymentRefNumber() {
        return paymentRefNumber;
    }
    public void setPaymentRefNumber(String paymentRefNumber) {
        this.paymentRefNumber = paymentRefNumber;
    }

    // A constructor method receiving all the arguments required to create each payment, including a reference to the corresponding expense object.
public Payment(Expense expense,String paymentRefNumber, String Name, String Number, String paymentDate, double paymentAmount) {
    this.expense = expense;
    this.Name = Name;
```

Ethan Couch
U79532295
IDE - IntelliJ

```
    this.Number = Number;
    this.paymentDate = paymentDate;
    this.paymentAmount = paymentAmount;
    this.paymentRefNumber= paymentRefNumber;
    expense.setPayment(this);

}
// An abstract print method printing the values of all variables. You may add
additional printing methods as needed.
public abstract void printMethod(); {

}


}

5
10,Electricity,March 1 2021,142.5,April 2 2021
11,Music,February 4 2021,10.0,March 4 2021
12,Cellular Data,January 28 2021,7.0,March 1 2021
13,Gas,January 29 2021,22.19,February 29 2021
14,Internet,January 17 2021,76.42,

5
10,Debit,Suntrust,Couch,AC85260,5487,March 28 2021,100.0
11,Credit,USF Visa,Couch,BD34235,A0100,February 5 2021,10.0
12,Credit,Mastercard,Couch,BD34235,3467,February 3 2021,7.0
13,Debit,Suntrust,Couch,AC85260,14629,February 2 2021,15.37
14,Debit,Bank of America,Couch,AC85260,I8527,January 29 2021,56.49
package payments;
//The Print interface is stored in package payments and contains:
public interface Print {
    // •A public print method.
    void printMethod();
}
```

Program Output:

"C:\Program Files\Java\jdk-14.0.2\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2020.2.1\lib\idea_rt.jar=25337:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2020.2.1\bin" -Dfile.encoding=UTF-8 -classpath C:\Assignment3_EthanCouch_U79532295\out\production\Assignment3 expenses.MyExpenses

Expenses for Ethan Couch

Ethan Couch
U79532295
IDE - IntelliJ

Expense Reference Number: 10
ServiceType: Electricity
BillDate: March 1 2021
BillAmount: 142.5
DueDate: April 2 2021
BalanceAmount: 42.5

Expense Reference Number: 11
ServiceType: Music
BillDate: February 4 2021
BillAmount: 10.0
DueDate: March 4 2021
BalanceAmount: 0.0

Expense Reference Number: 12
ServiceType: Cellular Data
BillDate: January 28 2021
BillAmount: 7.0
DueDate: March 1 2021
BalanceAmount: 0.0

Expense Reference Number: 13
ServiceType: Gas
BillDate: January 29 2021
BillAmount: 22.19
DueDate: February 29 2021
BalanceAmount: 6.820000000000002

Expense Reference Number: 14
ServiceType: Internet
BillDate: January 17 2021
BillAmount: 76.42
DueDate: February 18 2021
BalanceAmount: 19.93

Payments for Ethan Couch

Expense Reference Number: 10
Payment Reference Number: 5487
CreditCard: Suntrust
Card Name: Couch
Card Number: AC85260
Payment Date: March 28 2021
Payment Amount: 100.0

Ethan Couch
U79532295
IDE - IntelliJ

Expense Reference Number: 11
Payment Reference Number: A0100
CreditCard: USF Visa
Card Name: Couch
Card Number: BD34235
Payment Date: February 5 2021
Payment Amount: 10.0

Expense Reference Number: 12
Payment Reference Number: 3467
CreditCard: Mastercard
Card Name: Couch
Card Number: BD34235
Payment Date: February 3 2021
Payment Amount: 7.0

Expense Reference Number: 13
Payment Reference Number: 14629
CreditCard: Suntrust
Card Name: Couch
Card Number: AC85260
Payment Date: February 2 2021
Payment Amount: 15.37

Expense Reference Number: 14
Payment Reference Number: I8527
CreditCard: Bank of America
Card Name: Couch
Card Number: AC85260
Payment Date: January 29 2021
Payment Amount: 56.49

Total Expenses: 258.11
Total Balance: 69.25

Conclusions

After completing this assignment, I have created a program that is fully workable in the way of calculating total monthly expenses and balance, after the factorization of payments to each of these frequently occurring monthly expenses, in which these expenses and their respective applied payments are derived from two text files named ‘expenses.txt’ and ‘payments.txt’

Ethan Couch
U79532295
IDE - IntelliJ

respectively. I structured my code at large to be similar to my submission for assignment 2, with the addition of proper adjustments for assignment 3. Expense.java, Payment.java, CreditPayment.java, and DebitPayment.java, would be accessed by expense objects and payment objects as needed; I ensured that I established a setter and getter method for each of these data fields, where appropriate. I then created my constructor for each payment object, and modified the preexisting constructor for expense objects that I had from the last assignment. Additionally, I modified the constructors for CreditPayment and DebitPayment classes. For each of the aforementioned classes, I created a print method, where necessary, that prints the values of all variables, prefaced with the proper corresponding text as required by the sample program output. Then in a PrintAttributes() method, I printed some required text for the assignment, followed by 2 for loops that searches through each object in the *myExpenses* and *myPayments* arrays, calling the print methods that correspond to each object (printEx(); for each object in *myExpenses*, the print interface print method for each object in *myPayments*, which break down to credit and debit payments) that would then print the text and values of all respective variables.Lastly, I set variables totalExpense and totalBalance equal to the sum of each arrayed expense object's bill and balance and printed these variables out with their corresponding text. Assignment 3 saw its substantial differentiation with the requirement of me implementing a read() method that would parse through myExpenses and myPayments to read in the specific ordered parameters for the construction of each expense or payment object. Notably, another method, 'interact()' had to be implemented in which created a scanner object that would prompt the user for expenseRefNumber, and then enable the printing of the relevant expense or payment object.Ultimately, the main method called ObjectCreate() and PrintAttributes() and an interact() to produce the program's results, tempered by the ExpenseRefNumber that was read in by scanner input from the user. The greatest complication I faced in this assignment, was figuring out how to properly correlate each individual payment object with each expense object, given that both needed to be read in from text files, in order to pass the necessary information into the expense() and Credit/DebitPayment constructors to formulate a deduction of payments from expenses without hardcoded constructor values. Additionally, I faced considerable difficulty figuring out how to properly delineate each class's modifier, so that all methods could interact with each other as needed, such as was mandated by the changing of the Payments.java print method to an abstract method.