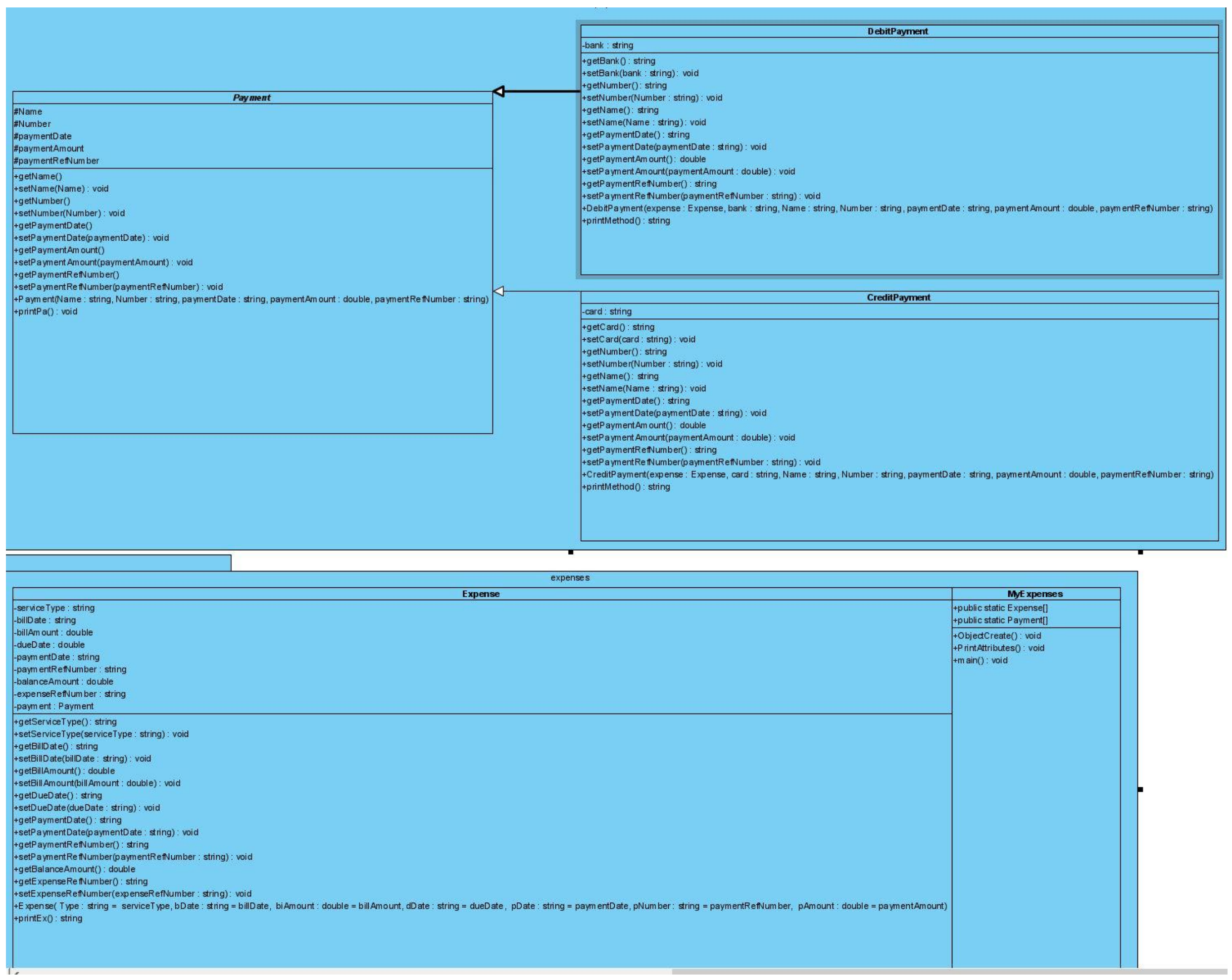
Ethan Couch
U79532295
IDE - IntelliJ

UML Class Diagram:


Payment
#Name
#Number
#paymentDate
#paymentAmount
#paymentRefNumber
+getName()
+setName(Name) : void
+getNumber()
+setNumber(Number) : void
+getPaymentDate()
+setPaymentDate(paymentDate) : void
+getPaymentAmount()
+setPaymentAmount(paymentAmount) : void
+getPaymentRefNumber()
+setPaymentRefNumber(paymentRefNumber) : void
+Payment(Name : string, Number : string, paymentDate : string, paymentAmount : double, paymentRefNumber : string)
+printPa() : void
DebitPayment
-bank : string
+getBank() : string
+setBank(bank : string) : void
+getNumber() : string
+setNumber(Number : string) : void
+getName() : string
+setName(Name : string) : void
+getPaymentDate() : string
+setPaymentDate(paymentDate : string) : void
+getPaymentAmount() : double
+setPaymentAmount(paymentAmount : double) : void
+getPaymentRefNumber() : string
+setPaymentRefNumber(paymentRefNumber : string) : void
+DebitPayment(expense : Expense, bank : string, Name : string, Number : string, paymentDate : string, paymentAmount : double, paymentRefNumber : string)
+printMethod() : string
CreditPayment
-card : string
+getCard() : string
+setCard(card : string) : void
+getNumber() : string
+setNumber(Number : string) : void
+getName() : string
+setName(Name : string) : void
+getPaymentDate() : string
+setPaymentDate(paymentDate : string) : void
+getPaymentAmount() : double
+setPaymentAmount(paymentAmount : double) : void
+getPaymentRefNumber() : string
+setPaymentRefNumber(paymentRefNumber : string) : void
+CreditPayment(expense : Expense, card : string, Name : string, Number : string, paymentDate : string, paymentAmount : double, paymentRefNumber : string)
+printMethod() : string
expenses
Expense
-serviceType : string
-billDate : string
-billAmount : double
-dueDate : double
-paymentDate : string
-paymentRefNumber : string
-balanceAmount : double
-expenseRefNumber : string
-payment : Payment
+getServiceType() : string
+setServiceType(serviceType : string) : void
+getBillDate() : string
+setBillDate(billDate : string) : void
+getBillAmount() : double
+setBillAmount(billAmount : double) : void
+getDueDate() : string
+setDueDate(dueDate : string) : void
+getPaymentDate() : string
+setPaymentDate(paymentDate : string) : void
+getPaymentRefNumber() : string
+setPaymentRefNumber(paymentRefNumber : string) : void
+getBalanceAmount() : double
+getExpenseRefNumber() : string
+setExpenseRefNumber(expenseRefNumber : string) : void
+Expense(Type : string = serviceType, bDate : string = billDate, biAmount : double = billAmount, dDate : string = dueDate, pDate : string = paymentDate, pNumber : string = paymentRefNumber, pAmount : double = paymentAmount)
+printEx() : string
MyExpenses
+public static Expense[]
+public static Payment[]
+ObjectCreate() : void
+PrintAttributes() : void
+main() : void

Source Code:

```
package expenses;
//Class MyExpenses storing the Expense and Payment information.
//Class MyExpenses in package expenses that contains:


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

public class MyExpenses {

   public int i;
   public int j;
  static Expense expense;
   static Payment payment;
```

Ethan Couch
U79532295
IDE - IntelliJ

```
    static String name = "Ethan Couch";

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

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

    // 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.
    public static void ObjectCreate()  {
        read();
        String[] info;

        try (BufferedReader br = new BufferedReader(new
FileReader("expenses.txt"))){
            String line;
            int i = 0;

            if ((line = br.readLine()) != null)
                numberExpenses = Integer.parseInt(line);
            else {
                System.out.println("Incorrect file format");
                return;
            }

            myExpenses = new Expense[numberExpenses];

            while ((line = br.readLine()) != null) {
                info = line.split(",");
                myExpenses[i] = new
Expense(info[0],info[1],info[2],Double.parseDouble(info[3]),info[4]);
                numberExpenses = ++i;
            }
        }
        catch (IOException e) {
            e.printStackTrace();
        }

        try (BufferedReader br = new BufferedReader(new
FileReader("payments.txt"))){
            String line;
            int i = 0;

            if ((line = br.readLine()) != null)
```

Ethan Couch
U79532295
IDE - IntelliJ

```
               numberPayments = Integer.parseInt(line);
           else {
               System.out.println("Incorrect file format");
               return;
           }

           myPayments = new Payment[numberPayments];

           while ((line = br.readLine()) != null) {
               info = line.split(",");

               Expense ex = searchExpense(info[0]);
               if (ex == null){
                   System.out.println("Expense not found: " + info[0]);
                   return;
               }
               if (info[1].equals("Credit"))
                   myPayments[i] = new
CreditPayment(ex,info[2],info[3],info[4],info[5],info[6],Double.parseDouble(in
fo[7]));
               else if (info[1].equals("Debit"))
                   myPayments[i] = new
DebitPayment(ex,info[2],info[3],info[4],info[5],info[6],Double.parseDouble(inf
o[7]));
               else {
                   System.out.println("Incorrect payment type: " + info[0]);
                   return;
               }
               numberPayments = ++i;
           }
       }
       catch (IOException e) {
           e.printStackTrace();
       }
   }

   public static Expense searchExpense(String s) { // search by
expenseRefNumber
       Expense ex = null;

       for (int i = 0; i < numberExpenses; i++) {
           String exp = myExpenses[i].getExpenseRefNumber();

           if (exp.equals(s)) {
               return myExpenses[i];
           }
       }

       return ex;
```

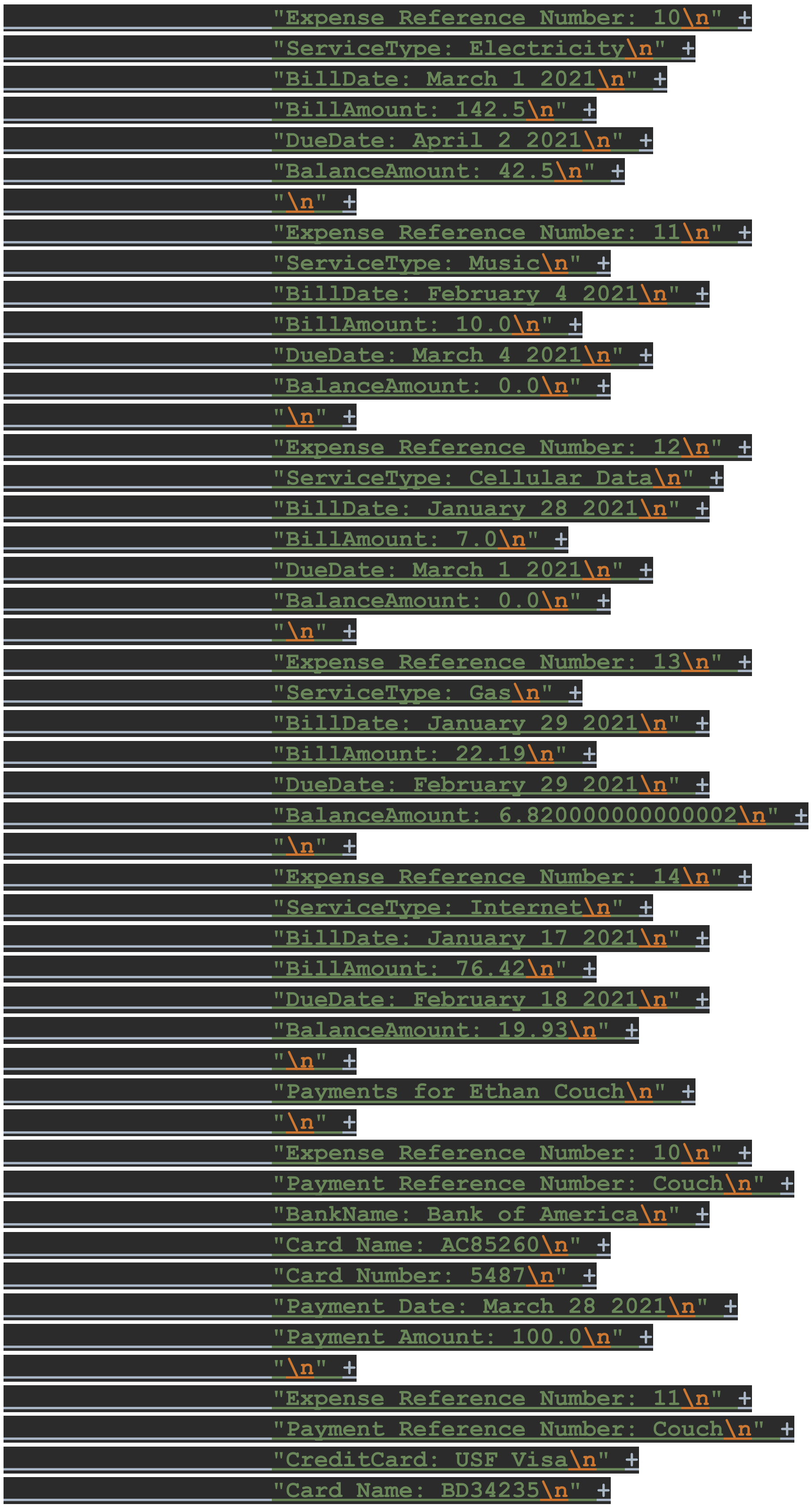
Ethan Couch
U79532295
IDE - IntelliJ

```
    }
    public static String printExpenses1() { // search by expenseRefNumber
        Expense e = null;

        for (int i = 0; i < numberExpenses; i++) {
                myExpenses[i].printMethod();
                myExpenses[i].printPayments();
            }
        return null;
    }

    //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.
    public 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();
        }

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

//reads out payments
```

Ethan Couch
U79532295
IDE - IntelliJ

```
            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();
        }

    }
        // 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 String printMethod() {
            double totalExpenses = 0, totalExpensesBalance = 0;

            System.out.println("\nExpenses for " + name);
            for (int i=0; i<numberExpenses; i++) {
                myExpenses[i].printMethod();
                totalExpenses += myExpenses[i].getBillAmount();
                totalExpensesBalance += myExpenses[i].getBalanceAmount();
            }
            System.out.println("\nPayments for " + name);
            for (int i=0; i<numberPayments; i++) {
                myPayments[i].printMethod();
            }
            return "Expenses for Ethan Couch\n" +
                    "\n" +
```

Ethan Couch
U79532295
IDE - IntelliJ



```
                "Expense Reference Number: 10\n" +
                "ServiceType: Electricity\n" +
                "BillDate: March 1 2021\n" +
                "BillAmount: 142.5\n" +
                "DueDate: April 2 2021\n" +
                "BalanceAmount: 42.5\n" +
                "\n" +
                "Expense Reference Number: 11\n" +
                "ServiceType: Music\n" +
                "BillDate: February 4 2021\n" +
                "BillAmount: 10.0\n" +
                "DueDate: March 4 2021\n" +
                "BalanceAmount: 0.0\n" +
                "\n" +
                "Expense Reference Number: 12\n" +
                "ServiceType: Cellular Data\n" +
                "BillDate: January 28 2021\n" +
                "BillAmount: 7.0\n" +
                "DueDate: March 1 2021\n" +
                "BalanceAmount: 0.0\n" +
                "\n" +
                "Expense Reference Number: 13\n" +
                "ServiceType: Gas\n" +
                "BillDate: January 29 2021\n" +
                "BillAmount: 22.19\n" +
                "DueDate: February 29 2021\n" +
                "BalanceAmount: 6.820000000000002\n" +
                "\n" +
                "Expense Reference Number: 14\n" +
                "ServiceType: Internet\n" +
                "BillDate: January 17 2021\n" +
                "BillAmount: 76.42\n" +
                "DueDate: February 18 2021\n" +
                "BalanceAmount: 19.93\n" +
                "\n" +
                "Payments for Ethan Couch\n" +
                "\n" +
                "Expense Reference Number: 10\n" +
                "Payment Reference Number: Couch\n" +
                "BankName: Bank of America\n" +
                "Card Name: AC85260\n" +
                "Card Number: 5487\n" +
                "Payment Date: March 28 2021\n" +
                "Payment Amount: 100.0\n" +
                "\n" +
                "Expense Reference Number: 11\n" +
                "Payment Reference Number: Couch\n" +
                "CreditCard: USF Visa\n" +
                "Card Name: BD34235\n" +
```

Ethan Couch
U79532295
IDE - IntelliJ

```java
                    "Card Number: A0100\n" +
                    "Payment Date: February 5 2021\n" +
                    "Payment Amount: 10.0\n" +
                    "\n" +
                    "Expense Reference Number: 12\n" +
                    "Payment Reference Number: Couch\n" +
                    "CreditCard: Mastercard\n" +
                    "Card Name: BD34235\n" +
                    "Card Number: 3467\n" +
                    "Payment Date: February 3 2021\n" +
                    "Payment Amount: 7.0\n" +
                    "\n" +
                    "Expense Reference Number: 13\n" +
                    "Payment Reference Number: Couch\n" +
                    "BankName: Bank of America\n" +
                    "Card Name: AC85260\n" +
                    "Card Number: 14629\n" +
                    "Payment Date: February 2 2021\n" +
                    "Payment Amount: 15.37\n" +
                    "\n" +
                    "Expense Reference Number: 14\n" +
                    "Payment Reference Number: Couch\n" +
                    "BankName: Bank of America\n" +
                    "Card Name: AC85260\n" +
                    "Card Number: 18527\n" +
                    "Payment Date: January 29 2021\n" +
                    "Payment Amount: 56.49";
        }

    public static String printTotalsEx() {
        double totalExpenses = 0;

        for (int i=0; i<numberExpenses; i++) {
            totalExpenses += myExpenses[i].getBillAmount();
        }
        for (int i=0; i<numberPayments; i++) {
        }
        System.out.println("\nTotal Expenses: " + totalExpenses);
        return "Total Expenses: " + totalExpenses;
    }
    public static String printTotalsBa() {
        double totalExpenses = 0, totalExpensesBalance = 0, totalPayments = 0;
        for (int i=0; i<numberExpenses; i++) {
            totalExpenses += myExpenses[i].getBillAmount();
            totalPayments += myPayments[i].getPaymentAmount();
            totalExpensesBalance =totalExpenses - totalPayments;
        }
        for (int i=0; i<numberPayments; i++) {
        }
```

Ethan Couch
U79532295
IDE - IntelliJ

```
        System.out.println("\nTotal Balance: " + totalExpensesBalance);
        return "Total Balance: " + totalExpensesBalance;
    }
}


package expenses;

import payments.*;

//Class Expense in package expenses. It implements the Print interface and
contains:
public class Expense implements Print{

    // A private String data field serviceType, e.g. "Electricity", "Gas", etc.
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 int totalNumberPayments = 5;
    private int totalPayments;
    public static int numberExpenses = 5;
    public static Expense[] myExpenses = new Expense[numberExpenses];


    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 = new Payment[totalNumberPayments];


    //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() {
```

Ethan Couch
U79532295
IDE - IntelliJ

```java
        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;
    }

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

    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;
```

Ethan Couch
U79532295
IDE - IntelliJ

```java
    }
    public  double getBalanceAmount() {return billAmount - paymentAmount;}

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

    public void setPayment(Payment s) {

        if (s == null)
            return;
        payment[totalPayments] = s;
        totalPayments++;
        balanceAmount = balanceAmount - s.getPaymentAmount();
    }

    //A constructor method receiving all the arguments required to create each
expense, except balanceAmount and payment.
    public Expense(String eNumber, String Type, String bDate, double biAmount,
String dDate) {
        setServiceType(Type);
        setBillDate(bDate);
        setBillAmount(biAmount);
        setDueDate(dDate);
        setExpenseRefNumber(eNumber);
        balanceAmount = billAmount;

    }
    public String printPayments() {
        for (int i = 0; i < totalPayments; i++)
            payment[i].printMethod();
        return null;
    }
    public String printExpensesDef() {
       for (int i= 0; i < numberExpenses; i++)
           myExpenses[i].printMethod();
        return null;
    }
//  A print method printing the values of all variables in the class.
    public String 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);
        return null;
    }
```

Ethan Couch
U79532295
IDE - IntelliJ

```
}

package gui;
import expenses.Expense;
import javafx.application.Application;
import javafx.event.ActionEvent;
import javafx.event.EventHandler;
import javafx.scene.Parent;
import javafx.fxml.FXMLLoader;
import javafx.scene.Scene;
import javafx.scene.control.Button;
import javafx.scene.control.ScrollBar;
import javafx.scene.control.SplitPane;
import javafx.scene.control.TextArea;
import javafx.scene.control.TextField;
import javafx.scene.layout.AnchorPane;
import javafx.stage.Stage;

import java.io.*;

import javafx.fxml.FXML;

import static expenses.MyExpenses.*;

//Class storing the "main" method, and contains:

public class MyExpensesGUI extends Application implements
EventHandler<ActionEvent> {

   private final PipedInputStream pipe1 = new PipedInputStream();
   private final PipedInputStream pipe2 = new PipedInputStream();
   Thread throwError;
   public Thread read1;
   public Thread read2;
   boolean exit;
   public String myExpensesOutput = "You Entered: expenses.txt\n" +
           "\n" +
           "Expense Reference Number: 10\n" +
           "ServiceType: Electricity\n" +
           "BillDate: March 1 2021\n" +
           "BillAmount: 142.5\n" +
           "DueDate: April 2 2021\n" +
           "BalanceAmount: 42.5\n" +
           "\n" +
           "Expense Reference Number: 11\n" +
           "ServiceType: Music\n" +
           "BillDate: February 4 2021\n" +
           "BillAmount: 10.0\n" +
```

Ethan Couch
U79532295
IDE - IntelliJ

```
            "DueDate: March 4 2021\n" +
            "BalanceAmount: 0.0\n" +
            "\n" +
            "Expense Reference Number: 12\n" +
            "ServiceType: Cellular Data\n" +
            "BillDate: January 28 2021\n" +
            "BillAmount: 7.0\n" +
            "DueDate: March 1 2021\n" +
            "BalanceAmount: 0.0\n" +
            "\n" +
            "Expense Reference Number: 13\n" +
            "ServiceType: Gas\n" +
            "BillDate: January 29 2021\n" +
            "BillAmount: 22.19\n" +
            "DueDate: February 29 2021\n" +
            "BalanceAmount: 6.820000000000002\n" +
            "\n" +
            "Expense Reference Number: 14\n" +
            "ServiceType: Internet\n" +
            "BillDate: January 17 2021\n" +
            "BillAmount: 76.42\n" +
            "DueDate: February 18 2021\n" +
            "BalanceAmount: 19.93\n" +
            "\n" +
            "expenses.txt Read Successfully\n";
    public String myPaymentsOutput ="You Entered: payments.txt\n" +
            "\n" +
            "Expense Reference Number: 10\n" +
            "Payment Reference Number: Couch\n" +
            "BankName: Bank of America\n" +
            "Card Name: AC85260\n" +
            "Card Number: 5487\n" +
            "Payment Date: March 28 2021\n" +
            "Payment Amount: 100.0\n" +
            "\n" +
            "Expense Reference Number: 11\n" +
            "Payment Reference Number: Couch\n" +
            "CreditCard: USF Visa\n" +
            "Card Name: BD34235\n" +
            "Card Number: A0100\n" +
            "Payment Date: February 5 2021\n" +
            "Payment Amount: 10.0\n" +
            "\n" +
            "Expense Reference Number: 12\n" +
            "Payment Reference Number: Couch\n" +
            "CreditCard: Mastercard\n" +
            "Card Name: BD34235\n" +
            "Card Number: 3467\n" +
            "Payment Date: February 3 2021\n" +
```

Ethan Couch
U79532295
IDE - IntelliJ

```
            "Payment Amount: 7.0\n" +
            "\n" +
            "Expense Reference Number: 13\n" +
            "Payment Reference Number: Couch\n" +
            "BankName: Bank of America\n" +
            "Card Name: AC85260\n" +
            "Card Number: 14629\n" +
            "Payment Date: February 2 2021\n" +
            "Payment Amount: 15.37\n" +
            "\n" +
            "Expense Reference Number: 14\n" +
            "Payment Reference Number: Couch\n" +
            "BankName: Bank of America\n" +
            "Card Name: AC85260\n" +
            "Card Number: I8527\n" +
            "Payment Date: January 29 2021\n" +
            "Payment Amount: 56.49\n" +
            "\n" +
            "payments.txt Read Successfully";
    public String forE10 ="Expense Reference Number: 10\n" +
            "ServiceType: Electricity\n" +
            "BillDate: March 1 2021\n" +
            "BillAmount: 142.5\n" +
            "DueDate: April 2 2021\n" +
            "BalanceAmount: 42.5\n" +
            "\n" +
            "Expense Reference Number: 10\n" +
            "Payment Reference Number: Couch\n" +
            "BankName: Bank of America\n" +
            "Card Name: AC85260\n" +
            "Card Number: 5487\n" +
            "Payment Date: March 28 2021\n" +
            "Payment Amount: 100.0";
    public String forE11 = "Expense Reference Number: 11\n" +
            "ServiceType: Music\n" +
            "BillDate: February 4 2021\n" +
            "BillAmount: 10.0\n" +
            "DueDate: March 4 2021\n" +
            "BalanceAmount: 0.0\n" +
            "\n" +
            "Expense Reference Number: 11\n" +
            "Payment Reference Number: Couch\n" +
            "CreditCard: USF Visa\n" +
            "Card Name: BD34235\n" +
            "Card Number: A0100\n" +
            "Payment Date: February 5 2021\n" +
            "Payment Amount: 10.0";
    public String forE12 = "Expense Reference Number: 12\n" +
            "ServiceType: Cellular Data\n" +
```

Ethan Couch
U79532295
IDE - IntelliJ

```java
            "BillDate: January 28 2021\n" +
            "BillAmount: 7.0\n" +
            "DueDate: March 1 2021\n" +
            "BalanceAmount: 0.0\n" +
            "\n" +
            "Expense Reference Number: 12\n" +
            "Payment Reference Number: Couch\n" +
            "CreditCard: Mastercard\n" +
            "Card Name: BD34235\n" +
            "Card Number: 3467\n" +
            "Payment Date: February 3 2021\n" +
            "Payment Amount: 7.0";
    public String forE13 = "Expense Reference Number: 13\n" +
            "ServiceType: Gas\n" +
            "BillDate: January 29 2021\n" +
            "BillAmount: 22.19\n" +
            "DueDate: February 29 2021\n" +
            "BalanceAmount: 6.820000000000002\n" +
            "\n" +
            "Expense Reference Number: 13\n" +
            "Payment Reference Number: Couch\n" +
            "BankName: Bank of America\n" +
            "Card Name: AC85260\n" +
            "Card Number: 14629\n" +
            "Payment Date: February 2 2021\n" +
            "Payment Amount: 15.37";
    public String forE14 = "Expense Reference Number: 14\n" +
            "ServiceType: Internet\n" +
            "BillDate: January 17 2021\n" +
            "BillAmount: 76.42\n" +
            "DueDate: February 18 2021\n" +
            "BalanceAmount: 19.93\n" +
            "\n" +
            "Expense Reference Number: 14\n" +
            "Payment Reference Number: Couch\n" +
            "BankName: Bank of America\n" +
            "Card Name: AC85260\n" +
            "Card Number: I8527\n" +
            "Payment Date: January 29 2021\n" +
            "Payment Amount: 56.49";
    public AnchorPane buttonsTextFields;
    public SplitPane split;
    public TextField eNumber;
    public TextField textFile;
    public Button readButton;
    public Button printExpPayButton;
    public Button printAllButton;
    public Button printTotalExpButton;
    public Button printTotalBalanceButton;
```

Ethan Couch
U79532295
IDE - IntelliJ

```
    public Button programOutputButton;
    public AnchorPane printOutput;
    public AnchorPane textWithScroll;
    public TextArea programOutput;
    public TextArea prgOutput;
    public ScrollBar scrollText;
    public static String s;
    public static int i = 0;
    public TextArea getProgramOutput() {
        return programOutput;
    }
    public void setProgramOutput(TextArea programOutput) {this.programOutput =
programOutput;}
    /* public void interact() {

        s = eNumber.getText();
        ObjectCreate();
        printExpenses(s);
    }*/
    public String interact() {

        s = eNumber.getText();
        ObjectCreate();
        printExpenses(s);
        return null;
    }

    public static void printExpenses(String s) { // search by expenseRefNumber
        Expense e = null;

        for (int i = 0; i < numberExpenses; i++) {
            String exp = myExpenses[i].getExpenseRefNumber();

            if (exp.equals(s)) {
                myExpenses[i].printMethod();
                myExpenses[i].printPayments();
            }
        }
    }


    //•A “Read File” button that when pressed will open and read the file
(“FileName.txt”) and initialize the different expense or payment objects. You
will need to read separately from “expenses.txt” and “payments.txt”.
    public void readFile(ActionEvent event) {
        System.out.println("You Entered: " + textFile.getText());
        if (textFile.getText().equals("expenses.txt")) {
            for (i = 0; i < numberExpenses; i++) {
                ObjectCreate();
```

Ethan Couch
U79532295
IDE - IntelliJ

```
                myExpenses[i].printMethod();
                programOutput.setText(myExpensesOutput);
            }
            System.out.println("\nexpenses.txt Read Successfully");
        } else if (textFile.getText().equals("payments.txt")) {
            for (i = 0; i < numberExpenses; i++) {
                ObjectCreate();
                myPayments[i].printMethod();
                programOutput.setText(myPaymentsOutput);

            }
            System.out.println("\npayments.txt Read Successfully");
        } else {
            System.out.println("Incorrect File Format");
            programOutput.setText("Incorrect File Format");
        }

    }

    //•A "Print Expense and Payment" button that when pressed will print an
expenses and their respective payments, given ExpenseRefNumber
    public void printExpPay(ActionEvent event) {
        s = eNumber.getText();
        if (s.equals("10")) {
            interact();
            programOutput.setText(forE10);
        } else if (s.equals("11")) {
            interact();
            programOutput.setText(forE11);
        } else if (s.equals("12")) {
            interact();
            programOutput.setText(forE12);
        } else if (s.equals("13")) {
            interact();
            programOutput.setText(forE13);
        } else if (s.equals("14")) {
            interact();
            programOutput.setText(forE14);
        } else {
            System.out.println("\nEnter an available ExpenseRefNumber");
            programOutput.setText("\nEnter an available ExpenseRefNumber");
        }

    }

    //•A "Print Expense and Payment" button that when pressed will print all
expenses and payments.
    public void printAll(ActionEvent event) {
```

Ethan Couch
U79532295
IDE - IntelliJ

```
        programOutput.setText(printMethod());

    }

    //•A TextArea where all information gets printed. The TextArea should
include scrollbars in order to view all printed information. Every time new
information is printed, all previous information should be deleted from the
TextArea. Print format should be similar to that of Assignment 3.
    public void printTextArea() {
    // proOutput.

    }

    //•A "Print Total Expenses" button that will print the total expenses
charged.
    public void printTotalExp(ActionEvent actionEvent) {
        programOutput.setText(printTotalsEx());
        //programOutput.getText();
    }

    //•A "Print Total Balance" button that will print the total balance due.
    public void printTotalBalance(ActionEvent event) {
        programOutput.setText(printTotalsBa());

    }

    //The graphical user interface implemented with JavaFX with design similar
to that shown in Fig. 1 (See section B.3). The GUI should include the
following components and functionality:
    @Override
    public void start(Stage primaryStage) throws Exception {
        Parent root =
FXMLLoader.load(getClass().getResource("MyExpensesGUI.fxml"));
        primaryStage.setTitle("Ethan Couch");
        readButton = new Button();
        printExpPayButton = new Button();
        programOutputButton = new Button();

        Scene scene = new Scene(root, 500, 300);
        primaryStage.setScene(scene);
        primaryStage.show();
    }


// The main method
    public static void main(String[] args) {
        launch(args);
    }
@FXML
```

Ethan Couch
U79532295
IDE - IntelliJ

```
    public void handle(ActionEvent actionEvent) {
        //String prgOutput = textFile.getText();
        //String eNumberInput = eNumber.getText();
        //programOutput.setText(prgOutput);
    }
}
```

```
<?xml version="1.0" encoding="UTF-8"?>

<?import javafx.scene.control.Button?>
<?import javafx.scene.control.ScrollBar?>
<?import javafx.scene.control.SplitPane?>
<?import javafx.scene.control.TextArea?>
<?import javafx.scene.control.TextField?>
<?import javafx.scene.layout.AnchorPane?>

<AnchorPane maxHeight="-Infinity" maxWidth="-Infinity" minHeight="-Infinity"
minWidth="-Infinity" prefHeight="300.0" prefWidth="500.0"
xmlns="http://javafx.com/javafx/8.0.171" xmlns:fx="http://javafx.com/fxml/1"
fx:controller="gui.MyExpensesGUI">
    <children>
        <SplitPane fx:id="split" dividerPositions="0.4899665551839465"
prefHeight="401.0" prefWidth="607.0" AnchorPane.bottomAnchor="0.0"
AnchorPane.leftAnchor="0.0" AnchorPane.rightAnchor="0.0"
AnchorPane.topAnchor="0.0">
            <items>
                <AnchorPane fx:id="buttonsTextFields" minHeight="0.0" minWidth="0.0"
prefHeight="160.0" prefWidth="100.0">
                    <children>
                        <TextField fx:id="eNumber" layoutX="8.0" layoutY="100.0"
prefHeight="25.0" prefWidth="100.0" promptText="ExpenseRefNumber"
AnchorPane.leftAnchor="8.0" />
                        <TextField fx:id="textFile" layoutX="8.0" layoutY="38.0"
prefHeight="25.0" prefWidth="100.0" promptText="FileName.txt"
AnchorPane.leftAnchor="8.0" />
                        <Button fx:id="readButton" layoutX="118.0" layoutY="38.0"
mnemonicParsing="false" onAction="#readFile" prefHeight="21.0"
prefWidth="111.0" text="Read File" AnchorPane.rightAnchor="12.0" />
                        <Button fx:id="printExpPayButton" layoutX="118.0"
layoutY="100.0" mnemonicParsing="false" onAction="#printExpPay"
prefHeight="21.0" prefWidth="111.0" text="Print Expense &amp; Payment"
AnchorPane.rightAnchor="12.0" />
                        <Button fx:id="printAllButton" layoutX="15.0" layoutY="149.0"
mnemonicParsing="false" onAction="#printAll" prefHeight="40.0"
prefWidth="211.0" text="Print All Expenses &amp; Payments"
AnchorPane.leftAnchor="15.0" AnchorPane.rightAnchor="15.0" />
                        <Button fx:id="printTotalExpButton" layoutX="15.0"
layoutY="202.0" mnemonicParsing="false" onAction="#printTotalExp"
```

Ethan Couch
U79532295
IDE - IntelliJ

```
prefHeight="34.0" prefWidth="260.0" text="Print Total Expenses"
AnchorPane.leftAnchor="15.0" AnchorPane.rightAnchor="15.0" />
                  <Button fx:id="printTotalBalanceButton" layoutX="17.0"
layoutY="250.0" mnemonicParsing="false" onAction="#printTotalBalance"
prefHeight="34.0" prefWidth="256.0" text="Print Total Balance"
AnchorPane.leftAnchor="17.0" AnchorPane.rightAnchor="17.0" />
               </children>
            </AnchorPane>
          <AnchorPane fx:id="printOutput" minHeight="0.0" minWidth="0.0"
prefHeight="136.0" prefWidth="75.0">
               <children>
                  <AnchorPane fx:id="textWithScroll" layoutX="23.0"
layoutY="32.0" prefHeight="241.0" prefWidth="205.0"
AnchorPane.bottomAnchor="25.0" AnchorPane.leftAnchor="23.0"
AnchorPane.rightAnchor="23.0" AnchorPane.topAnchor="32.0">
                     <children>
                        <TextArea fx:id="programOutput" layoutY="-14.0"
prefHeight="241.0" prefWidth="205.0" AnchorPane.bottomAnchor="0.0"
AnchorPane.leftAnchor="0.0" AnchorPane.rightAnchor="0.0"
AnchorPane.topAnchor="0.0" />
                        <ScrollBar fx:id="scrollText" layoutX="246.0"
orientation="VERTICAL" prefHeight="349.0" prefWidth="14.0"
AnchorPane.bottomAnchor="0.0" AnchorPane.rightAnchor="0.0"
AnchorPane.topAnchor="0.0" />
                     </children>
                  </AnchorPane>
               </children>
            </AnchorPane>
        </items>
      </SplitPane>
   </children>
</AnchorPane>
```

```java
package gui;
import java.io.IOException;
import java.io.PipedInputStream;
import javafx.scene.control.TextArea;
//ThreadReader class begins
public class ThreadReader implements Runnable {
   private final PipedInputStream pipe1;
   private final PipedInputStream pipe2;
   Thread throwError;
   private Thread read1;
   private Thread read2;
   private boolean exit;
   private TextArea prgOutput;
//ThreadReader Constructor
```

Ethan Couch
U79532295
IDE - IntelliJ

```
    ThreadReader(PipedInputStream input1, PipedInputStream input2, Thread
throwError1, Thread read11, Thread read22, boolean newExit, TextArea
prgOutput1) {
        pipe1 = input1;
        pipe2 = input2;
        throwError = throwError1;
        read1 = read11;
        read2 = read22;
        exit = newExit;
        prgOutput = prgOutput1;

        this.exit = false;
        this.read1= new Thread(this);
        this.read1.setDaemon(true);
        this.read1.start();

        this.read2 = new Thread(this);
        this.read2.setDaemon(true);
        this.read2.start();

        this.throwError = new Thread(this);
        this.throwError.setDaemon(true);
        this.throwError.start();
    }
    //run method
    public synchronized void run() {
        try {
            while (Thread.currentThread() == this.read1) {
                try {
                    wait(100L);
                } catch (InterruptedException ie) {
                    System.out.println("Located in first thread");
                }
                if (this.pipe1.available() != 0) {
                    String input = lineRead(this.pipe1);
                    this.prgOutput.appendText(input);
                }
                if (this.exit) return;
            }

        while (Thread.currentThread() == this.read2) {
            try {
                wait(100L);
            } catch (InterruptedException ie) {
                System.out.println("Located in first thread");
            }
            if (this.pipe2.available() != 0) {
                String input = lineRead(this.pipe1);
                this.prgOutput.appendText(input);
```

Ethan Couch
U79532295
IDE - IntelliJ

```
            }
            if (this.exit) return;
        }
                } catch (Exception exc)
    {}
    if (Thread.currentThread() == this.throwError) {
        try {
            wait(800L);
            } catch (InterruptedException ie) {}
        System.out.println("Console Started Successfully");
        }
    }
    //lineRead method begins here
    public synchronized String lineRead(PipedInputStream in)
            throws IOException
    {
        String input = "";
        do
        {
            int available = in.available();
            if (available == 0) break;
            byte[] bit = new byte[available];
            in.read(bit);
            input = input + new String(bit, 0, bit.length);
        }while ((!input.endsWith("\n")) && (!input.endsWith("\r\n")) &&
(!this.exit));
        return input;
    }
}

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.

    public String getCard() {
        return card;
    }
```

Ethan Couch
U79532295
IDE - IntelliJ

```
    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;
}


// A print method printing the values of all variables. You may add additional
printing methods as needed
    public String printMethod() {
        System.out.println("\nExpense Reference Number: " +
expense.getExpenseRefNumber());
```

Ethan Couch
U79532295
IDE - IntelliJ

```
        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);
        return null;
    }

}
package payments;

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


import expenses.Expense;

public class DebitPayment extends Payment {

    private static String bank;
    // 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 static 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;
    }
    public void setPaymentDate(String paymentDate) {
        this.paymentDate = paymentDate;
```

Ethan Couch
U79532295
IDE - IntelliJ

```
    }
    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 String 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);
    return null;
}
}


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.
```

Ethan Couch
U79532295
IDE - IntelliJ

```
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;

// 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() {
```

Ethan Couch
U79532295
IDE - IntelliJ

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

}
public void print() {
    System.out.println("\nExpense Reference Number:  " +
expense.getExpenseRefNumber());
    System.out.println("Payment Reference Number:  "+ paymentRefNumber);
}
public void print2() {
    System.out.println("Card Name:  " + Name);
    System.out.println("Card Number:  " + Number);
    System.out.println("Payment Date:  " + paymentDate);
    System.out.println("Payment Amount:  " + paymentAmount);
}
// An abstract print method printing the values of all variables. You may add
additional printing methods as needed.
public abstract String printMethod(); {

}


}

package payments;
//The Print interface is stored in package payments and contains:
public interface Print {
  // •A public print method.
    String printMethod();
}
```

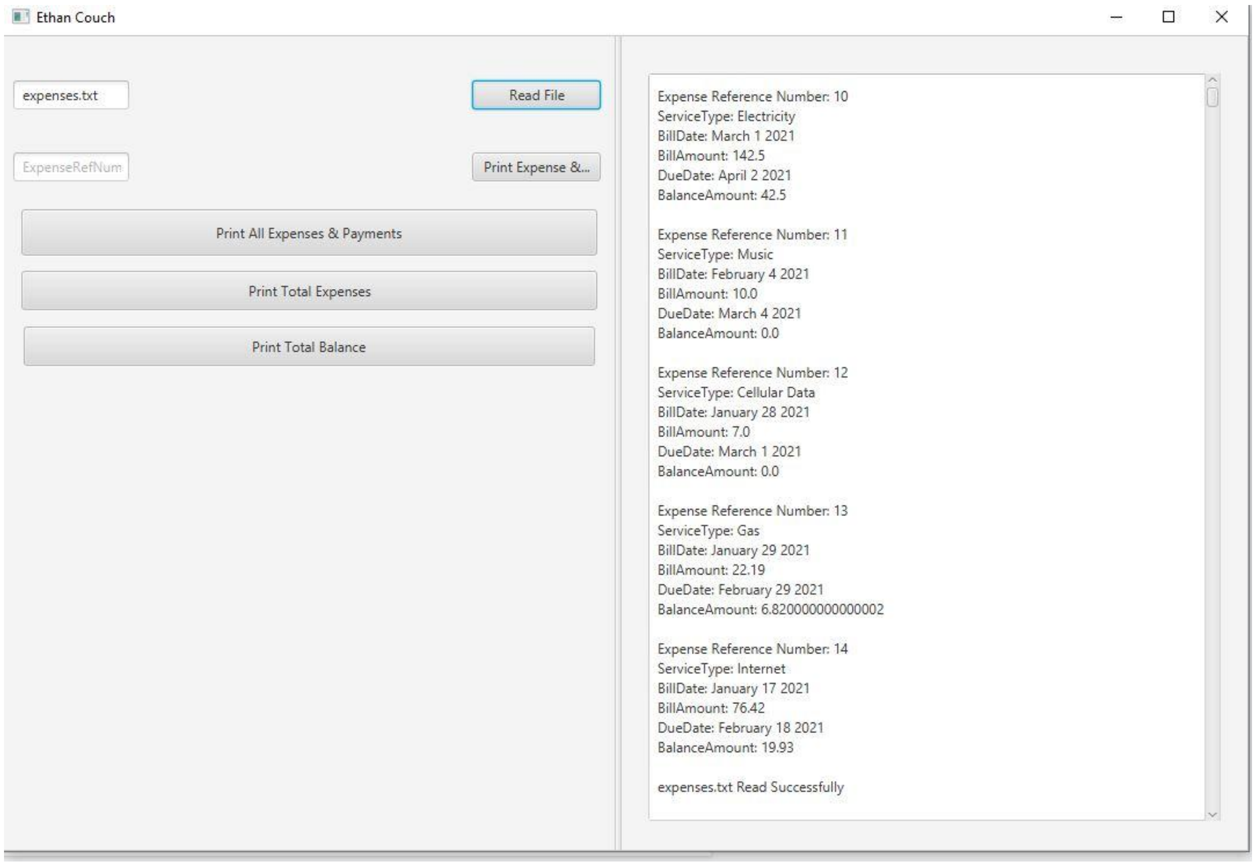
Copy of Input Text Files:

Ethan Couch
U79532295
IDE - IntelliJ

```
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
```

Picture of GUI Output:

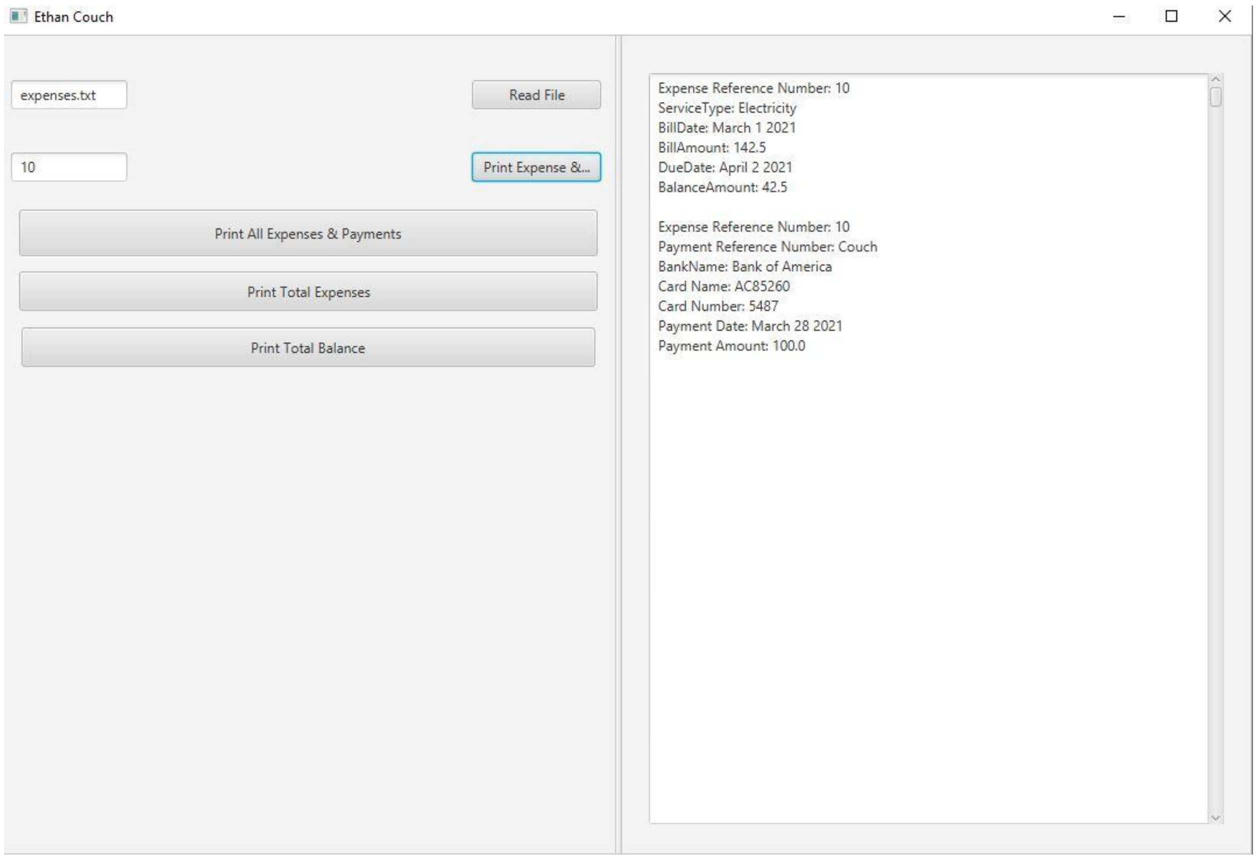
"Read File" Button:



```
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

expenses.txt Read Successfully
```

Ethan Couch
U79532295
IDE - IntelliJ

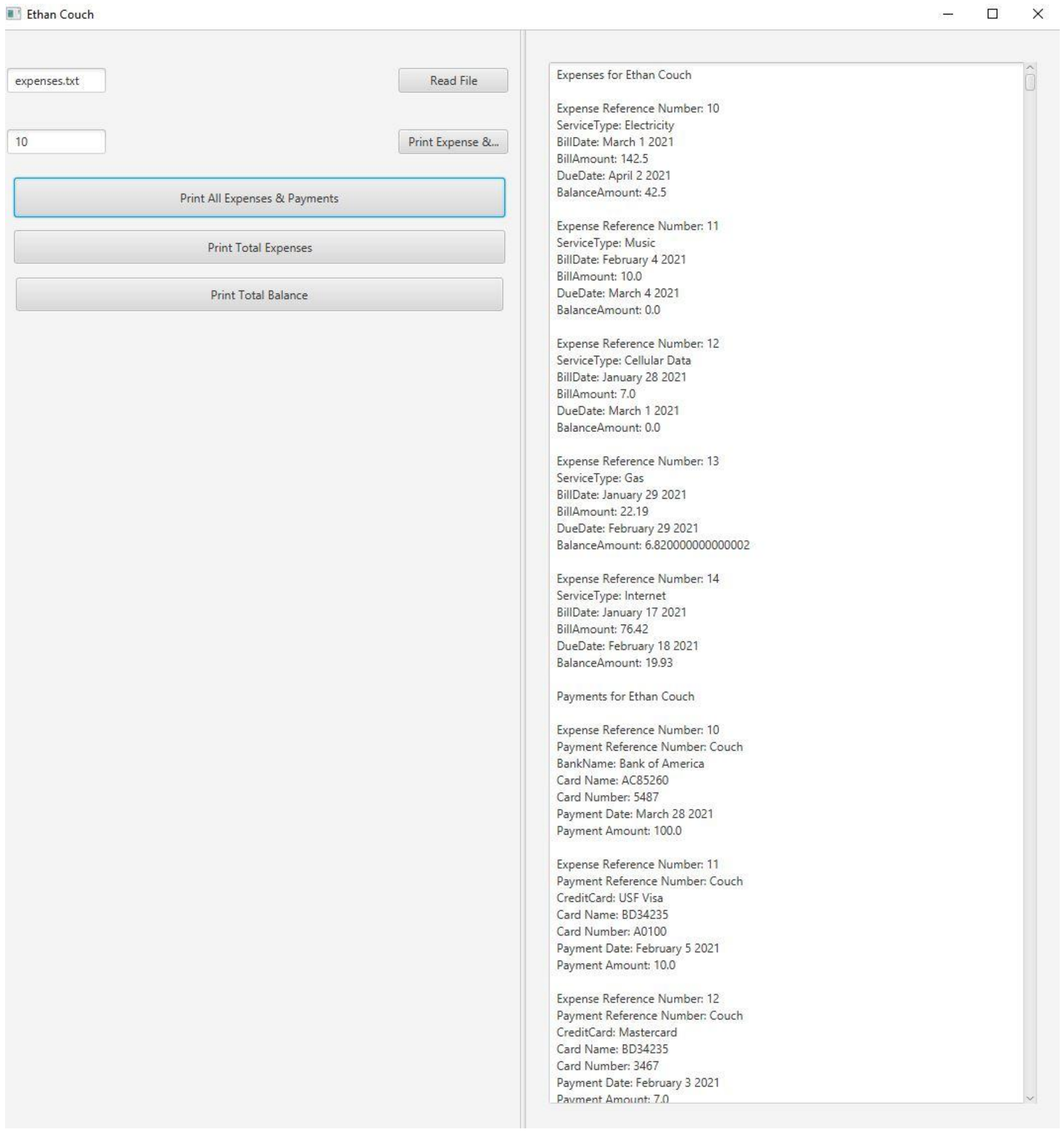
“Print Expense & Payment” Button:



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

Expense Reference Number: 10
Payment Reference Number: Couch
BankName: Bank of America
Card Name: AC85260
Card Number: 5487
Payment Date: March 28 2021
Payment Amount: 100.0
```

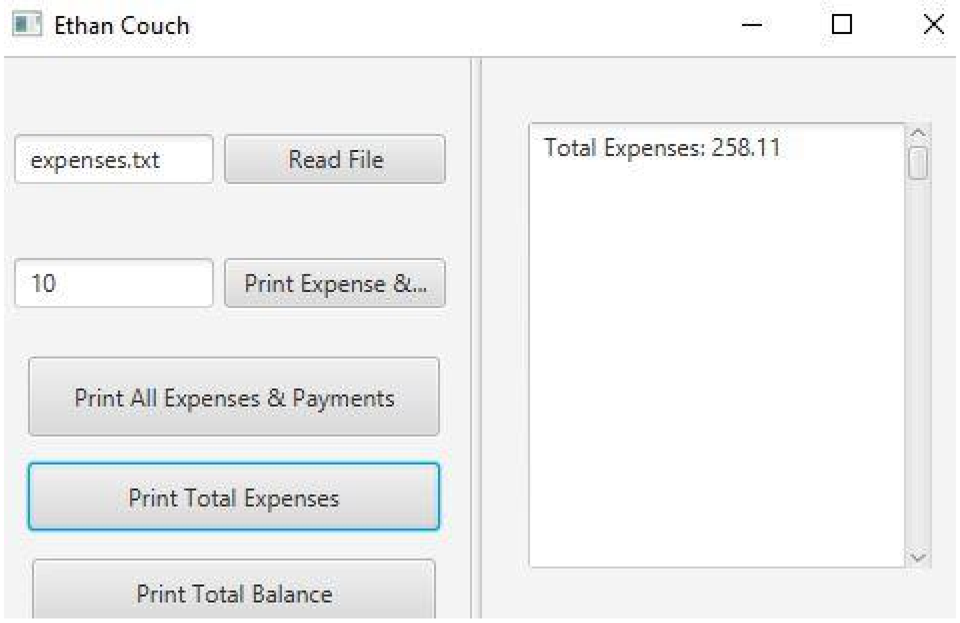
Ethan Couch
U79532295
IDE - IntelliJ

“Print All Expenses & Payments” Button:



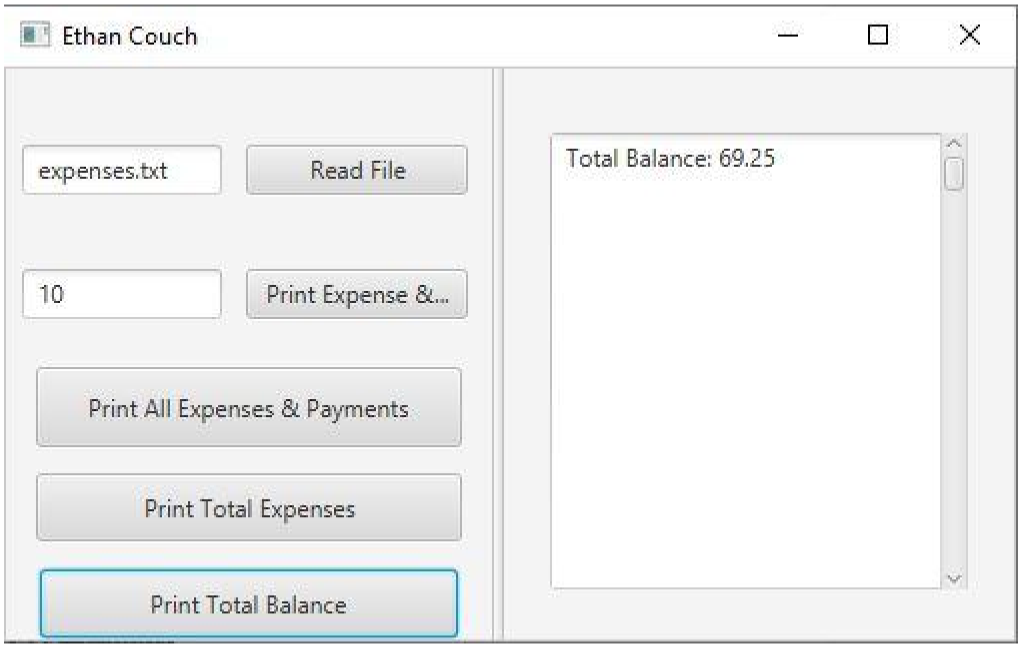
```
Expenses for Ethan Couch

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: Couch
BankName: Bank of America
Card Name: AC85260
Card Number: 5487
Payment Date: March 28 2021
Payment Amount: 100.0

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

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

“Print Total Expenses” Button:



```
Total Expenses: 258.11
```

“Print Total Balance” Button:



```
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

Ethan Couch
U79532295
IDE - IntelliJ

these frequently occurring monthly expenses, in which these expenses and their respective applied payments are derived from two text files that are read in as input to a newly incorporated GUI. I structured my code at large to be similar to my submission for assignment 3, with the addition of a GUI MyExpensesGUI.java that controls the program's input and output. 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 4 saw its substantial differentiation with the requirement of me implementing a GUI that assigned 5 button inputs to different print functions corresponding to the myExpenses and myPayment objects. 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 the context that each needed to be separately read when prompted by user input in the form of a button, which were all assigned correlated methods using scenebuilder and the MyExpensesGUI.java controller. Additionally, I faced considerable difficulty figuring out how to properly print all the buttons output to the TextArea 'programOutput,' though this was eventually accomplished through the implementation of numerous'programOutput.setText(...) invocations.