Software Architectures

ICT 373

Assignment 1 – 2020

Student: Faiz Syed

ID: 33243485

Professor: Ferdous Sohel

Question 1

a) Description of the problem:

The solution aims to use JavaScript and HTML to write a web page which collects the user's name, phone number (with any prefix codes), birth date, and favourite pastime (Surfing the Web, Playing Sport, Listening to Music, Watching TV, Playing Games, Community Service, Daydreaming, Reading, or Meditation only) and sends them off to a server. Some reasonable assumptions were made about the content and format of the input data and those assumptions were enforced from within your page using Run-time and Batch validation techniques.

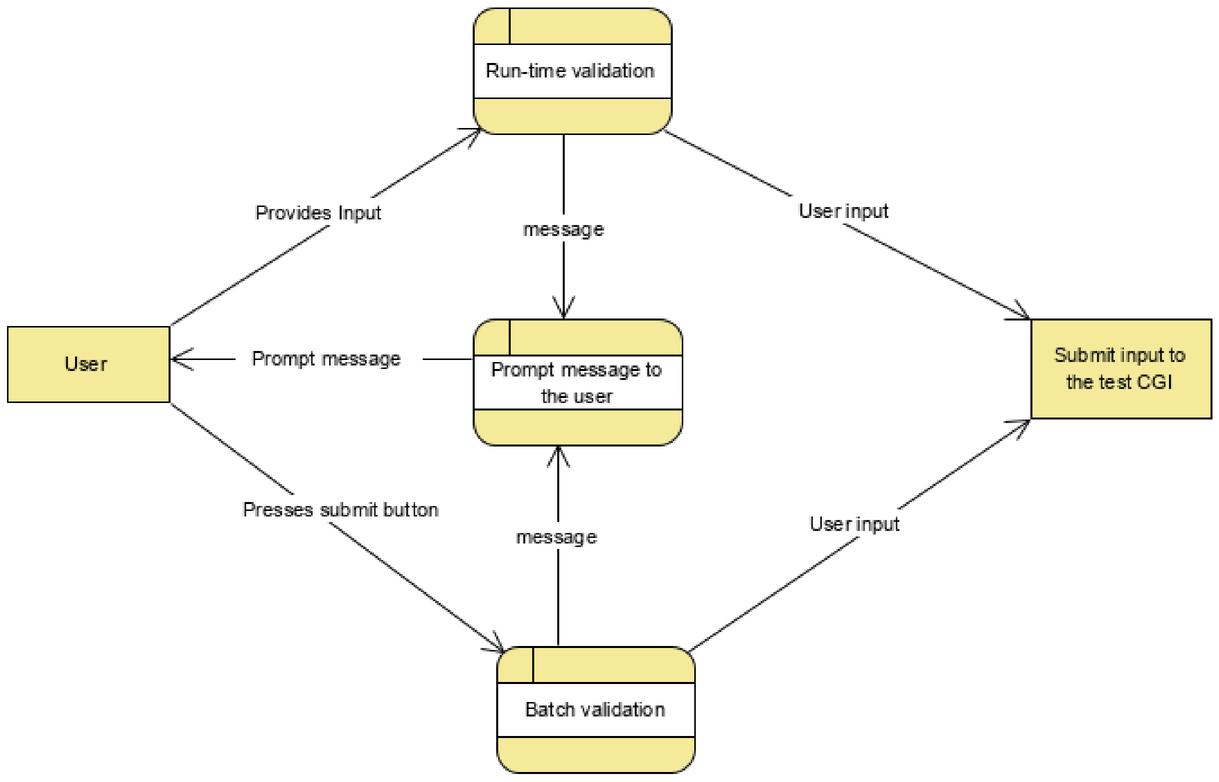
The address of the test CGI destination is http://www.it.murdoch.edu.au/cgi-bin/reply1.pl

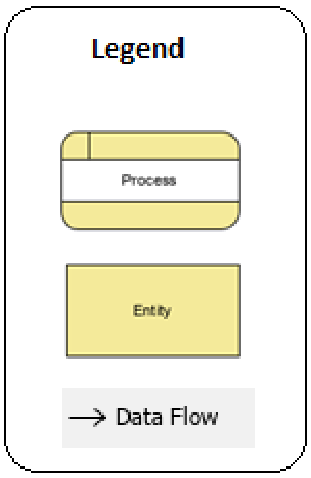
b) Solution Description (Design and Pseudo code):

Basic HTML form has been used to create the user input form, text inputs for username, date of birth and phone number, and a dropdown list for taking pastime input. When the user provides an input the onChange event is invoked which makes sure that the field is non-empty and contains valid input. If the input is invalid, then a message a prompted to the screen directing the user to input in the desired format. This type of validation is called run-time validation.

The JavaScript Date object has been used to get the current date and validate against the user input for dob to make sure the entered date is not in the future. When the user presses the submit button batch validation is triggered which only accepts the submission when all the input fields are validated. The input gets submitted to the specified test CGI.

Data Flow Diagram:





Pseudo code:

```
//Checks if the input box is empty
SUBMODULE: isNotEmpty
IMPORT: field
EXPORT: (boolean)
ALGORITHM:
    inputStr = (value of field)
    IF(inputStr = "" OR inputStr = null) THEN
        ALERT "This field required an entry."
        RETURN False
    END IF

    RETURN True


//Check if the user enters letters in the username field
SUBMODULE: isLetter
IMPORT: field
EXPORT: (boolean)
ALGORITHM:
    IF (field isNotEMPTY) THEN
        inputtxt = (value of field)
        letters = (A-Za-z)
        IF(inputtxtx MATCHES letters) THEN
            RETURN True
        ELSE
            ALERT "Please enter letters only!"
            RETURN FALSE
        END IF
    END IF


//Phone number validator
SUBMODULE: validatePhone
IMPORT: field
EXPORT: (boolean)
ALGORITHM:
    IF (field isNotEmpty) THEN
        phone  = (value of field)

        IF(phone MATCHES +44 123456789) THEN
```

```
            RETURN True
        ELSE
            ALERT "Invalid number! please use this format (+61 123456789)"
            RETURN False
        END IF
    END IF


//Date of birth validator
SUBMODULE: validateDOB
IMPORT: field
EXPORT: (boolean)
ALGORITHM:
    IF(field isNotEmpty) THEN
        myDate = current date
        dob = (value of field)

        IF(dob MATCHES dd/mm/yyyy) THEN
            IF(dob > MyDate) THEN
                ALERT "Please enter a date in the past!"
                RETURN False
            ELSE
                RETURN True
            END IF
        ELSE
            ALERT "Please follow this format (dd/mm/yyyy)"
            RETURN False
        END IF
    END IF
    RETURN False


//compares the present and given dates and returns true if the date is in
future
SUBMODULE: compareDates
IMPORT: date1(DATE), date2(DATE)
EXPORT: (boolean)
ALGORITHM:
    d1 = day FROM date1
    m1 = month FROM date1
    y1 = year FROM date1

    d2 = day FROM date2
    m2 = month FROM date2
    y2 = year FROM date2

    IF(y1 > y2) THEN
        RETURN True
    ELSE
        IF(y1 EQUALS y2 AND m1 EQUALS m2) THEN
            RETURN True
        ELSE
            IF(y1 EQUALS y1&& m1 EQUALS m2 && d1 > d2) THEN
                RETURN True
            END IF
        END IF
    END IF

    RETURN False
```

```
//form validator
SUBMODULE: validate
IMPORT: form
EXPORT: (boolean)
ALGORITHM:
    IF(username isLetter AND dob validateDOB AND phone validatePhone) THEN
        RETURN True
    ELSE
        RETURN FALSE
    END IF
```

c) Source code listing:

```
<html>
<head>
    <title>Assignment 1</title>
    <script language="JavaScript">

    //Checks if the input box is empty
    function isNotEmpty(field)
    {
        var inputStr = field.value
        if (inputStr == "" || inputStr == null ) {
            alert("This field required an entry.")
            field.focus()
            field.select()
            return false
        }
        return true
    }

    //Check if the user enters letters in the username field
    function isLetter(field)
    {
        if (isNotEmpty(field))
        {
            var inputtxt =  field.value;

            var letters = /^[A-Za-z ]+$/;
            if(inputtxt.match(letters))
            {
                return true;
            }
            else
            {
                alert("Please enter letters only!");
                return false;
            }
        }
        return false;
    }

    //Phone number validator
    function validatePhone(field)
    {
        if (isNotEmpty(field))
        {
            var phone  = field.value;;

            if(phone.match(/^\+?([0-9]{2})\)?[-. ]?([0-9]{4})[-. ]?([0-
9]{5})$/))
            {
                return true;
            }
            else
            {
                alert("Invalid number! please use this format (+61
123456789)");
                return false;
            }

        }
        return false;
    }


    //Date of birth validator
    function validateDOB(field)
    {
        if (isNotEmpty(field))
        {
            var MyDate = new Date();
            var MyDateString;

            MyDateString = ('0' + MyDate.getDate()).slice(-2) + '/'
                         + ('0' + (MyDate.getMonth()+1)).slice(-2) + '/'
                         + MyDate.getFullYear();


            var dob = field.value.toString();

            if(dob.match(/^(0?[1-9]|[12][0-9]|3[01])[\/\-](0?[1-
9]|1[012])[\/\-]\d{4}$/))
            {
                if(compareDates(dob, MyDateString))
                {
                    alert("Please enter a date in the past!");
                    return false;
                }
                else
                    return true;
            }
            else
            {
                alert("Please follow this format (dd/mm/yyyy)");
                return false;
            }
        }
        return false;
    }


    //compares the present and given dates and returns true if the date is
in future
    function compareDates(date1, date2)
    {
        var parts = date1.split('/');
        var date1 = (parts[0] + parts[1] + parts[2]);
        parts = date2.split('/');
        var date2 = parts[0] + parts[1] + parts[2];

        var d1 = Number(date1.slice(0,2));
        var m1 = Number(date1.slice(2,4));
        var y1 = Number(date1.slice(4,8));

        var d2 = Number(date2.slice(0,2));
        var m2 = Number(date2.slice(2,4));
        var y2 = Number(date2.slice(4,8));

        if(y1 > y2)
        {
            return true;
        }
        else
            if(y1 == y2 && m1 > m2)
            {
                return true;
            }
            else
                if(y1==y2 && m1==m2 && d1 > d2)
                {
                    return true;
                }

        return false;

    }

    //form validator
    function validate(form)
    {
        if  (isLetter(form.username) && validateDOB(form.dob) &&
validatePhone(form.phone))
            return true
        else
            return false

    }


</script>
</head>

<body>

    <form method="POST" action="http://www.it.murdoch.edu.au/cgi-
bin/reply1.pl"
                                onSubmit="return validate(this)"
style="background-color:#ffc">

    <br>
    <p>Username: <input id="username" name="username" type ="text" size
="10" onChange = "isLetter(this)" onclick = "this.select()"></p>

    <p>DOB: <input id="dob" name="dob" type="text" size="10"
onChange="validateDOB(this)" onclick = "this.select()"></p>
```

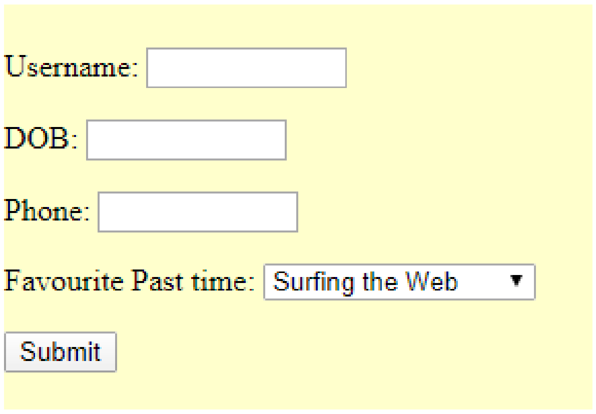
```
    <p>Phone: <input id="phone" name="phone" type="text" size="10"
onChange="validatePhone(this)" onclick = "this.select()"></p>

    <label for="pastime">Favourite Past time:</label>
    <select name="pastime" id="pastime">
    <option value="Surfing the Web">Surfing the Web</option>
    <option value="Playing Sport">Playing Sport</option>
    <option value="Listening to Music">Listening to Music</option>
    <option value="Watching TV">Watching TV</option>
    <option value="Playing Games">Playing Games</option>
    <option value="Community Service">Community Service</option>
    <option value="Daydreaming">Daydreaming</option>
    <option value="Reading">Reading</option>
    <option value="Meditation">Meditation</option>
    </select>

    <p><input type="submit"> </p><br>

    </form>

</body>
</html>
```

d) Testing and Sample results:

No.	Test	Expected result	Actual result	Pass/ Fail?
1	Loading page	Page should load properly	Page loads as expected	Pass
2	Input fields	Input fields should be presented	All input fields were presented as per the requirements	Pass
3	Preventing empty inputs from being submitted	A message should be prompted to the user	A message is prompted to the user	Pass
4	Providing invalid username	A message should be prompted	A message asking the user to enter letters only is prompted	Pass
5	Providing invalid date of birth	A message should be prompted	A message asking the user to input dob in (dd/mm/yyyy) format is presented	Pass
6	Proving invalid phone number	A message should be prompted	A message asking the user to input phone number in a particular format is prompted	Pass
7	Run-time checking	Check input for all input fields	All the fields as mentioned above have run-time validation	Pass
8	Form submission	Form should get submitted	Form gets submitted successfully	Pass

9	Batch validation	Perform batch validation on form submission	Batch validation is performed on submission	Pass
10	Field auto selection on mouse click	The filed should get highlighted automatically	The field gets highlighted automatically	Pass
11	Drop down list to get pastime input	User should be able to make selection form the dropdown list	User can make selection from the dropdown list	Pass
12	Avoiding use of date object	Find an alternative solution	No alternative solution was found, hence, for convenience the Date object had to be used	Fail
13	Use of regular expressions	Use regex appropriately	Regex have been appropriately used in the solution	Pass

- Input fields:



Username:

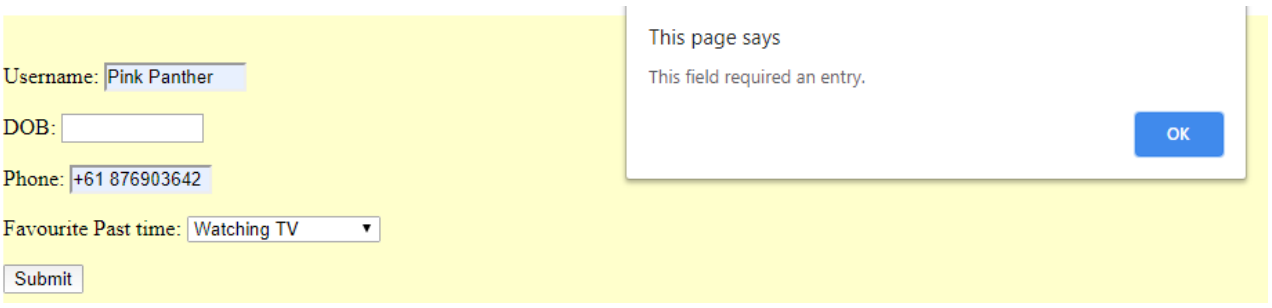
DOB:

Phone:

Favourite Past time: Surfing the Web

Result: web page successfully loads up

- Empty field



Username: Pink Panther

DOB:

Phone: +61 876903642

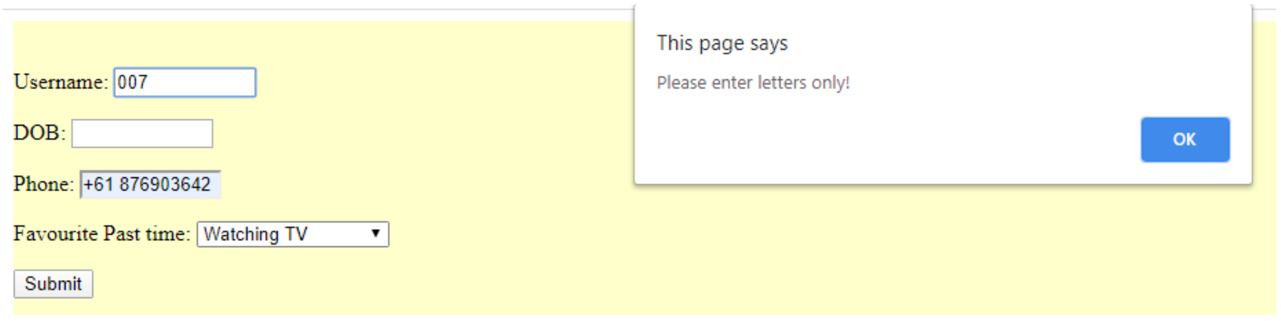
Favourite Past time: Watching TV

This page says

This field required an entry.

Result: An empty field is detected during run-time as well as during batch validation

- Invalid username



Username: 007

DOB:

Phone: +61 876903642

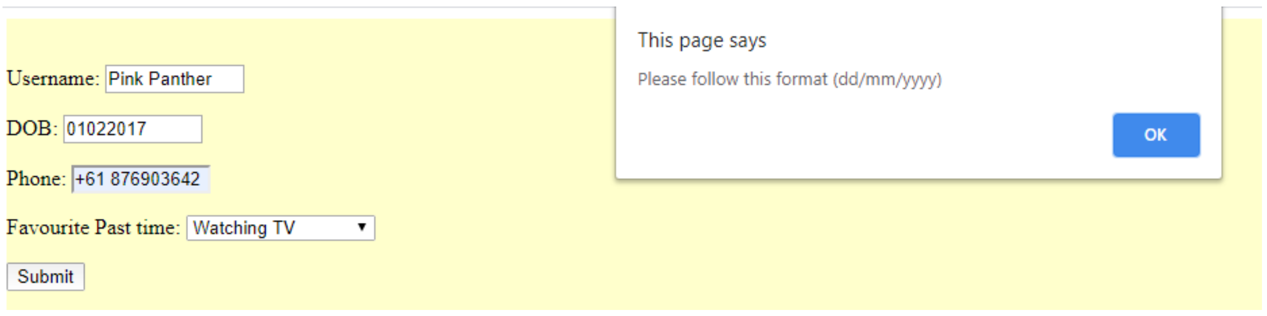
Favourite Past time: Watching TV

This page says

Please enter letters only!

Result: an error message is prompted

- Invalid date of birth:



Username: Pink Panther

DOB: 01022017

Phone: +61 876903642

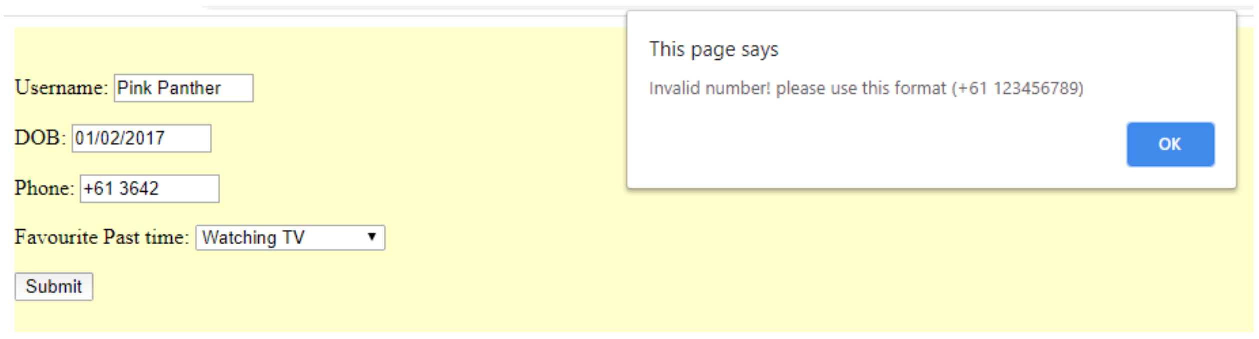
Favourite Past time: Watching TV

This page says

Please follow this format (dd/mm/yyyy)

Result: an error message is prompted

- Invalid phone number:



Username: Pink Panther

DOB: 01/02/2017

Phone: +61 3642

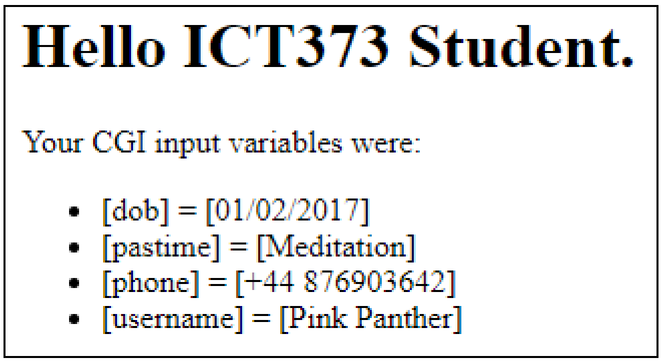
Favourite Past time: Watching TV

This page says

Invalid number! please use this format (+61 123456789)

Result: an error message is prompted

- Form submission



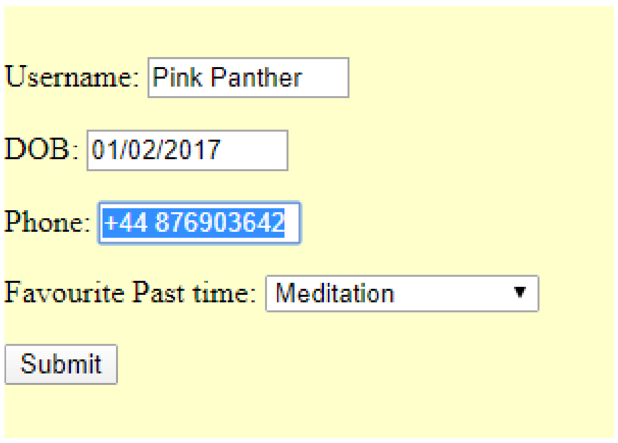
Hello ICT373 Student.

Your CGI input variables were:

- [dob] = [01/02/2017]
- [pastime] = [Meditation]
- [phone] = [+44 876903642]
- [username] = [Pink Panther]

Result: Form successfully submitted

- Auto selection on mouse click (extra feature):



Username:	Pink Panther
DOB:	01/02/2017
Phone:	+44 876903642
Favourite Past time:	Meditation

Result: The input text is highlight on mouse click

Question 2

Title:

Title: Weekly Magazine Service

Author: Faiz Syed

Date: 25th- March – 2020

File names: Magazine.java, Supplement.java, Customer.java, AssociateCustomer.java and Dates.java

Purpose: Program to manage an online weekly personalized magazine service

Requirements/Specifications:

Requirements:

Design, implement in Java, test and document a set of classes for use in a program to manage an online weekly personalized magazine service.

a) A customer has a name, an email address and a list of supplements which they are interested in.
b) A paying customer has a payment method and a list of associate customers whom they also pay for.
c) An associate customer is a customer whose subscription is paid for by a specified paying customer.
d) A payment method could be by a specified credit card or direct debit from a specified bank account
e) A supplement has a name and a weekly cost
f) The magazine also has a weekly cost for the main part
g) Customer gets a weekly subscription email listing the supplements that they subscribed to.
h) Paying customer gets a monthly email telling them how much is being to their card or account for the month and itemizing the cost by supplements for them and any of their associate customers

Specifications:

1) The magazine has a minimum of 3-4 supplements with made-up details
2) Creates 5-6 customers of different types with made-up details
3) Automatically adds there paying customers, namely:
   a. Tom Sawyer
   b. James Richard
   c. Julian Corman
4) Associate customers can be added to existing paying customers and the payment bills will be updated accordingly:
   a. Tom Sawyer – Alex, Peter
   b. James Richard –
   c. Julian Corman – John
5) All paying customers are stored in an ArrayList
6) Each paying customer maintains an ArrayList of its associate customers
7) Each customer has a separate ArrayList of supplements they have subscribed to

8) All customers are free to subscribe and unsubscribe supplements, and their payment bills will be adjusted accordingly
9) The program prints weekly subscription emails for the paying customers
10) The program prints monthly payment bill for paying customer, the bill also includes the cost of services used by their associate customers
11) Associate customers can be removed from the given paying customers list, and the payment bill will be adjusted accordingly
12) When a paying customer is removed then all its associate customers are also removed from the magazine service

USER GUIDE:

System requirements:

- ✓ Windows 7 (or later version) will be ideal to support this program

Compiler requirements:

- ✓ Java Compiler and utilities which can be download from the internet free of cost will be required.
- ✓ Java SE Development Kit 8 Update 22 (64-bit) need to be installed on the machine to run the program.
- ✓ Kindly follow this software installation instructions for the JDK: http://www.java.com/en/download/help/windows_manual_download.xml
- ✓ The following packages need to be installed:
    - o jdk-8u202-windows-x64.exe
    - o Java SE Runtime Environment 8u202 32-bit & 64-bit
    - o jre-8u202-windows-i586.exe
    - o jre-8u202-windows-x64.exe

Install NetBeans IDE:

- ✓ NetBeans IDE 8.0.2 can be used to run the program
- ✓ Install NetBeans 8.0.2 from https://netbeans.org/downloads/old/8.2/

Running the code:

- ✓ Open NetBeans (Make sure you set all the paths before this)
- ✓ Go to “File” and click on “Open Project”, then locate and select the “Magazine” project to load it
- ✓ Once the project “Magazine” is open in NetBeans, click Run Project (F6) to run this project.
- ✓ Output can be viewed in the output terminal, and Javadoc web page can be opened form the Javadoc terminal.

Structure/Design:

The program is built on an object-oriented design which includes use of different classes, objects, encapsulation and polymorphism. A Class is a blueprint for creating objects, and objects are instances of a class. Encapsulation is a mechanism for wrapping up of variables and method in a single unit. Using encapsulation variables and methods of a class can be hidden from other classes, and can only be accessed by the methods of the current class. The variables can be kept private and public setters and getters methods can be provided to modify or obtain the values.

Class description:

Magazine.java – It contains the main method and executes the main functionalities

Supplement.java – This class contains the information of different supplements

Customer.java – This class contains the information of paying customers

AssociateCustomer.java – This class contains the information of associate customers

Dates.java – This class computes the number of weeks in a given month

Approach:

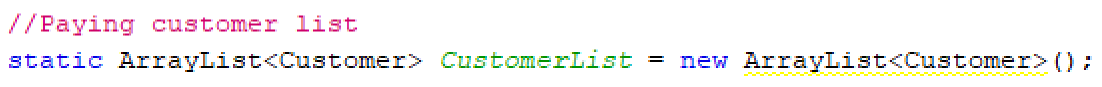
- The class Magazine.java is the main class of the program.

- The solution does not have two separate classes for customer and paying customer because there is no difference in properties. So, in this program the Customer.java class itself is the class for the paying customer, and can be easily reused as a generic customer class in the future.

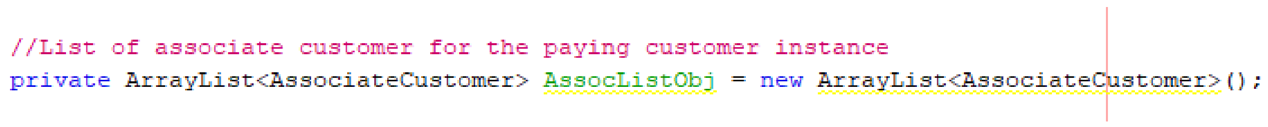
Use of Data structure:

The program uses ArrayLists to store customer information in the form of object, and form creating a list of supplements they have subscribed to. The alternative option to ArrayList was array.

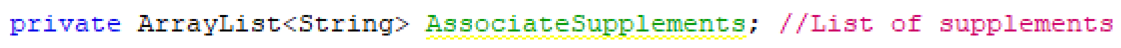
The ArrayList class is a resizable array, which can be found in the *java.util* package. The difference between a built-in array and an ArrayLIst in Java, is that the size of an array cannot be modified when required. Which means if we want to add or remove elements to/from an array, then a new array needs to be created. While elements can be added and removed from an ArrayList whenever we want.



```java
//Paying customer list
static ArrayList<Customer> CustomerList = new ArrayList<Customer>();
```

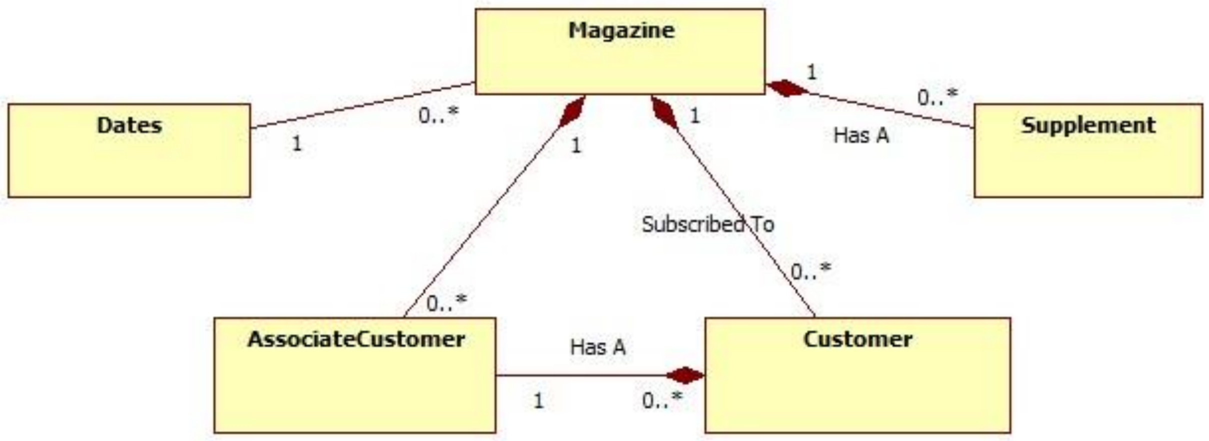


```java
//List of associate customer for the paying customer instance
private ArrayList<AssociateCustomer> AssocListObj = new ArrayList<AssociateCustomer>();
```

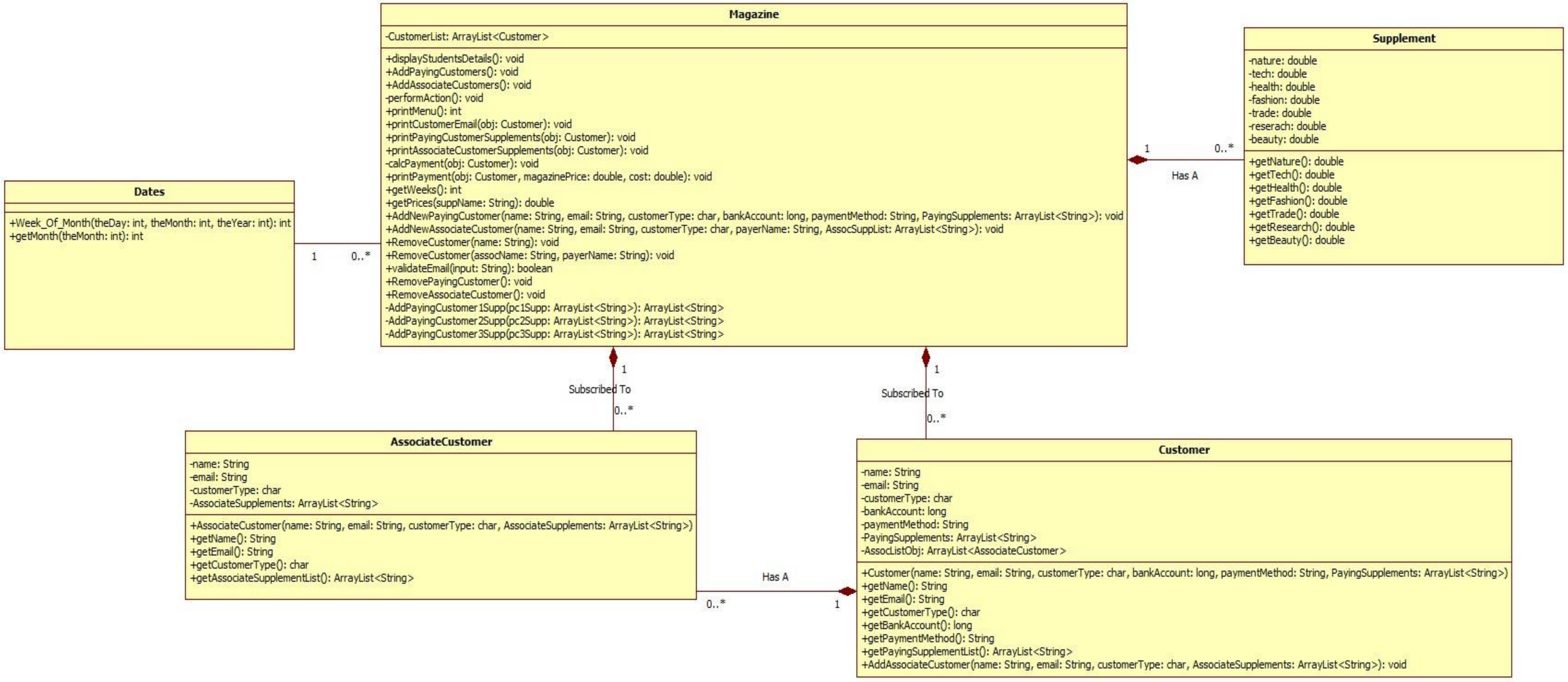


```java
private ArrayList<String> AssociateSupplements; //List of supplements
```

High-level UML diagram:



Low-level UML diagram:


Magazine
-CustomerList: ArrayList<Customer>
+displayStudentsDetails(): void
+AddPayingCustomers(): void
+AddAssociateCustomers(): void
-performAction(): void
+printMenu(): int
+printCustomerEmail(obj: Customer): void
+printPayingCustomerSupplements(obj: Customer): void
+printAssociateCustomerSupplements(obj: Customer): void
-calcPayment(obj: Customer): void
+printPayment(obj: Customer, magazinePrice: double, cost: double): void
+getWeeks(): int
+getPrices(suppName: String): double
+AddNewPayingCustomer(name: String, email: String, customerType: char, bankAccount: long, paymentMethod: String, PayingSupplements: ArrayList<String>): void
+AddNewAssociateCustomer(name: String, email: String, customerType: char, payerName: String, AssocSuppList: ArrayList<String>): void
+RemoveCustomer(name: String): void
+RemoveCustomer(assocName: String, payerName: String): void
+validateEmail(input: String): boolean
+RemovePayingCustomer(): void
+RemoveAssociateCustomer(): void
-AddPayingCustomer1Supp(pc1Supp: ArrayList<String>): ArrayList<String>
-AddPayingCustomer2Supp(pc2Supp: ArrayList<String>): ArrayList<String>
-AddPayingCustomer3Supp(pc3Supp: ArrayList<String>): ArrayList<String>
Supplement
-nature: double
-tech: double
-health: double
-fashion: double
-trade: double
-reserach: double
-beauty: double
+getNature(): double
+getTech(): double
+getHealth(): double
+getFashion(): double
+getTrade(): double
+getResearch(): double
+getBeauty(): double
1
0..*
Has A
Dates
+Week_Of_Month(theDay: int, theMonth: int, theYear: int): int
+getMonth(theMonth: int): int
1
0..*
1
Subscribed To
0..*
1
Subscribed To
0..*
AssociateCustomer
-name: String
-email: String
-customerType: char
-AssociateSupplements: ArrayList<String>
+AssociateCustomer(name: String, email: String, customerType: char, AssociateSupplements: ArrayList<String>)
+getName(): String
+getEmail(): String
+getCustomerType(): char
+getAssociateSupplementList(): ArrayList<String>
Has A
0..*
1
Customer
-name: String
-email: String
-customerType: char
-bankAccount: long
-paymentMethod: String
-PayingSupplements: ArrayList<String>
-AssocListObj: ArrayList<AssociateCustomer>
+Customer(name: String, email: String, customerType: char, bankAccount: long, paymentMethod: String, PayingSupplements: ArrayList<String>)
+getName(): String
+getEmail(): String
+getCustomerType(): char
+getBankAccount(): long
+getPaymentMethod(): String
+getPayingSupplementList(): ArrayList<String>
+AddAssociateCustomer(name: String, email: String, customerType: char, AssociateSupplements: ArrayList<String>): void

Pseudo code:

Magazine.java

```
SUBMODULE: calcPayment
IMPORT: obj (Customer)
EXPORT: (void)
ALGORITHM:
    suppName (String)
    numOfWeeks (Integer)

    CONSTRUCT sdf USING "MMM/yyyy"
    CONSTRUCT date USING Date()

    magazinePrice (double) = 2
    cost (double) = 0

    //Adding up cost of supplements subscribed by Paying Customer
    FOR i=0 TO obj.getPayingSupplementList.size <- none (exclusive) INC BY
1
        suppName = obj.getPayingSupplementList.get <- i
        cost = cost + getPrices <- suppName
    END FOR

    //Adding up cost of supplements subscribed by Associate Customer
    FOR i=0 TO obj.getAssocCustomer.size <- none (exclusive) INC BY 1
        FOR j=0 TO obj.getAssocCustomer <-
i.getAssociateSupplementList.size <- none (exclusive) INC BY 1
            suppName = obj.getAssocCustomer <-
i.getAssociateSupplementList.get <- java/date-time/java-date-examples/
            cost = cost + getPrices <- suppName
        END FOR
    END FOR

    //Adding up the fixed cost of magazine
    cost = cost + magazinePrice * (obj.getAssocCustomer.size <- none + 1)

    //To get number of weeks in the current month
    numOfWeeks = getWeeks <- none

    //cost times by the number of weeks in the month
    cost = cost * numOfWeeks

    //Output of total cost
    printPayment <- obj, magazinePrice, cost
```

Magazine.java

```
SUBMODULE: AddPayingCustomers
IMPORT: none
EXPORT: (void)
ALGORITHM:
      CONSTRUCT pc1Supp (ArrayList<String>) USING default
      pc1Supp = AddPayingCustomer1Supp <- pc1Supp
      AddNewPayingCustomer <- "Tom Sawyer", "tom@student.com", 'P',
1234567, "Debit", pc1Supp

      CONSTRUCT pc2Supp (ArrayList<String>) USING default
      pc2Supp = AddPayingCustomer2Supp <- pc2Supp
      AddNewPayingCustomer <- "James Richard", "james@student.com", 'P',
1947667, "Credit", pc2Supp

      CONSTRUCT pc3Supp (ArrayList<String>) USING default
      pc3Supp = AddPayingCustomer3Supp <- pc3Supp
      AddNewPayingCustomer <- "Julian Corman", "julie@student.com", 'P',
464564, "Debit", pc3Supp

      OUTPUT "No. of paying customer added: " + CustomerList.size <- none
      FOR int = 0 TO CustomerList.size <- none (exclusive) INC BY 1
            OUTPUT CustomerList.get <- i.getName <- none
      END FOR
```

Dates.java

```
SUBMODULE: Week_Of_Month
IMPORT: theDay (Integer), theMonth (Integer), theYear (Integer)
EXPORT: numOfWeeksInMonth (Integer)
ALGORITHM:
    CONSTRUCT calendar AS getInstance <- none
    year (Integer) = theYear
    month (Integer) getMonth <- theMonth
    day (Integer) = theDay
    calendar.set <- year, month, day

    numOfWeeksInMonth (Integer) = calendar.getActualMaximum <-
Calendar.WEEK_OF_MONTH

    OUTPUT "Number of Weeks In Month" + numOfWeeksInMonth

    RETURN numOfWeeksInMonth
```

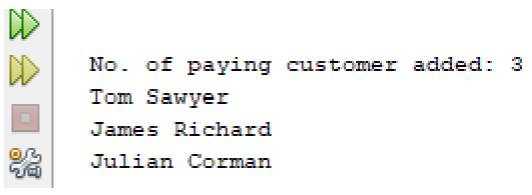
Limitation:

1) Code can be made more modular.
2) Weekly email is only displayed once because outputting 4 weekly magazine emails for each paying customer was not presentable as it looked repetitive.
3) Initially, the code was written to take user input from a menu. Later on, when the requirements had changed, the code was redesigned to simply display all the output without interacting with the user by using some hardcoded examples.
4) Exception handling was being put in place when there was requirement for user's input, but it was abandoned when that requirement was crossed out. So, more exception handling can be added.

Testing:

1) Adding Paying customers

Three paying customers are added:



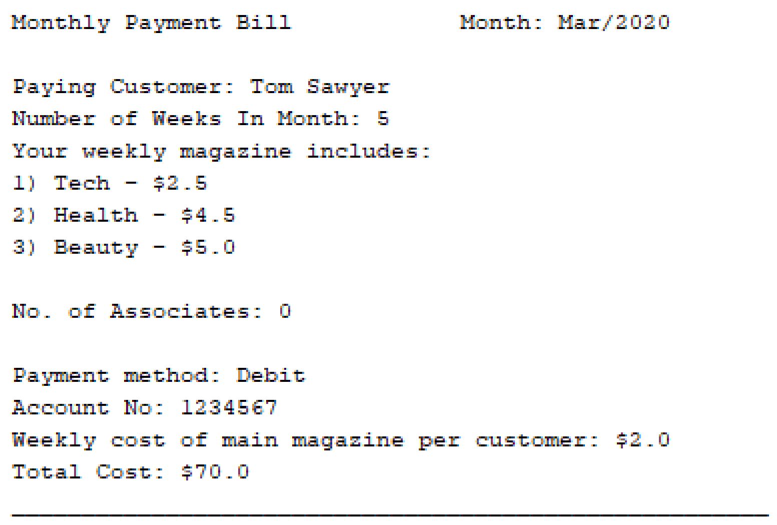
```
No. of paying customer added: 3
Tom Sawyer
James Richard
Julian Corman
```

Weekly subscription emails for these customers:

```
Dear Tom Sawyer,                2020-03-24
Your weekly magazine includes:
1) Tech - $2.5
2) Health - $4.5
3) Beauty - $5.0


Dear James Richard,             2020-03-24
Your weekly magazine includes:
1) Tech - $2.5


Dear Julian Corman,             2020-03-24
Your weekly magazine includes:
1) Research - $3.5
```

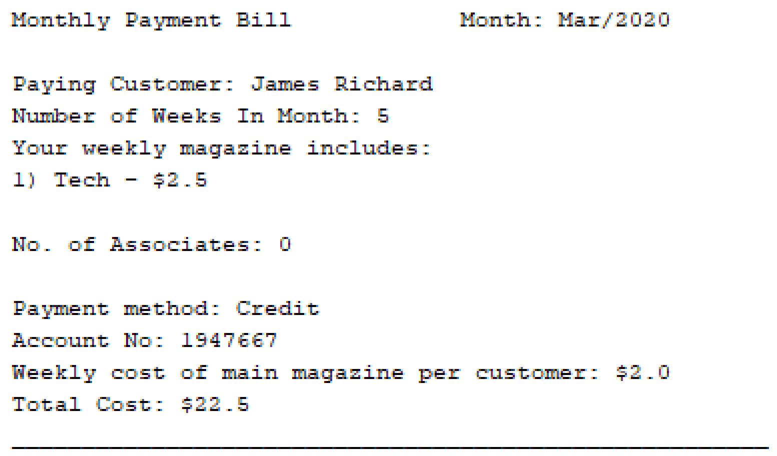
Monthly payment Bills for these customers:

(Tom Sawyer)



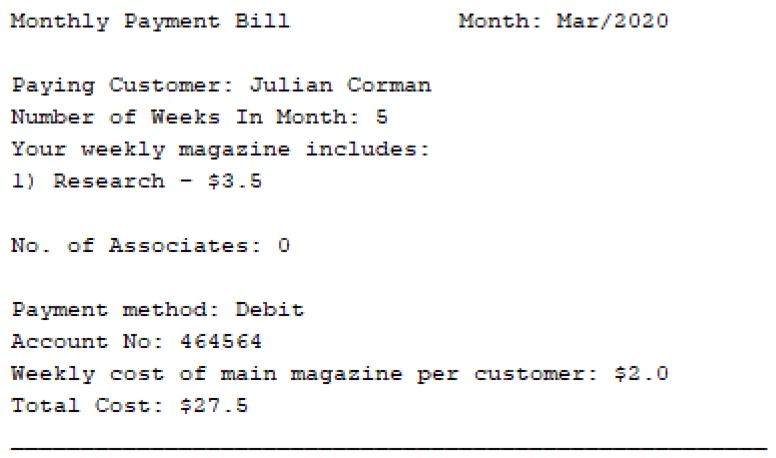
```
Monthly Payment Bill             Month: Mar/2020

Paying Customer: Tom Sawyer
Number of Weeks In Month: 5
Your weekly magazine includes:
1) Tech - $2.5
2) Health - $4.5
3) Beauty - $5.0

No. of Associates: 0

Payment method: Debit
Account No: 1234567
Weekly cost of main magazine per customer: $2.0
Total Cost: $70.0
```

(James Richard)



```
Monthly Payment Bill             Month: Mar/2020

Paying Customer: James Richard
Number of Weeks In Month: 5
Your weekly magazine includes:
1) Tech - $2.5

No. of Associates: 0

Payment method: Credit
Account No: 1947667
Weekly cost of main magazine per customer: $2.0
Total Cost: $22.5
```

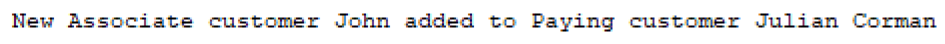
(Julian Corman)



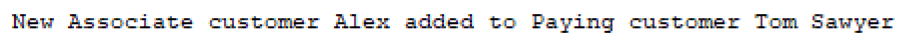
```
Monthly Payment Bill             Month: Mar/2020

Paying Customer: Julian Corman
Number of Weeks In Month: 5
Your weekly magazine includes:
1) Research - $3.5

No. of Associates: 0

Payment method: Debit
Account No: 464564
Weekly cost of main magazine per customer: $2.0
Total Cost: $27.5
```

2) Adding Associate customers

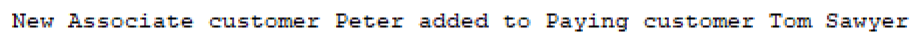
New associate customers were added to paying customer Julian Corman and Tom Sawyer:



```
New Associate customer John added to Paying customer Julian Corman
```



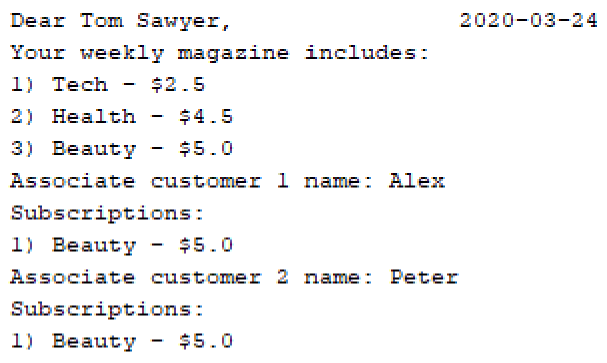
```
New Associate customer Alex added to Paying customer Tom Sawyer
```



```
New Associate customer Peter added to Paying customer Tom Sawyer
```

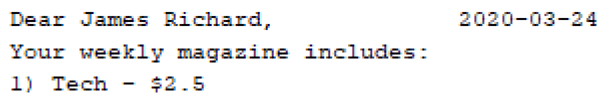
Weekly subscription email for the Paying customers after adding some associate customers:

(Tom Sawyer)



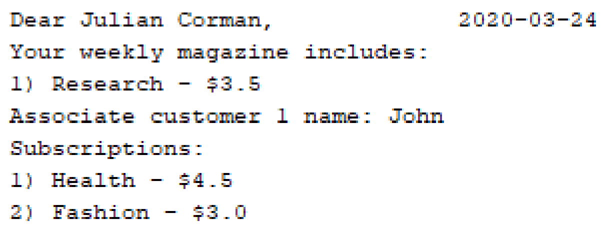
```
Dear Tom Sawyer,                2020-03-24
Your weekly magazine includes:
1) Tech - $2.5
2) Health - $4.5
3) Beauty - $5.0
Associate customer 1 name: Alex
Subscriptions:
1) Beauty - $5.0
Associate customer 2 name: Peter
Subscriptions:
1) Beauty - $5.0
```

(James Richard)



```
Dear James Richard,             2020-03-24
Your weekly magazine includes:
1) Tech - $2.5
```

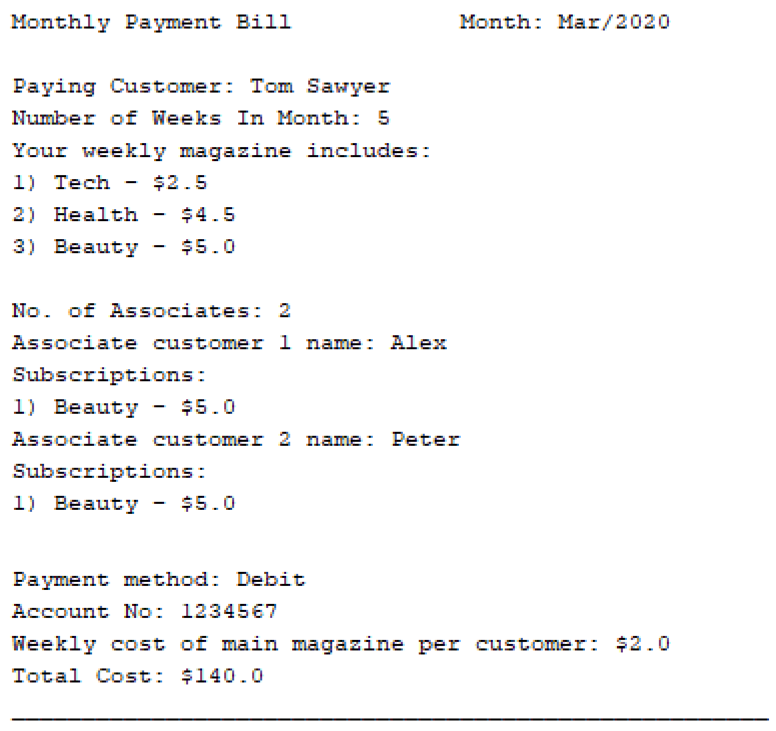
(Julian Corman)



```
Dear Julian Corman,             2020-03-24
Your weekly magazine includes:
1) Research - $3.5
Associate customer 1 name: John
Subscriptions:
1) Health - $4.5
2) Fashion - $3.0
```

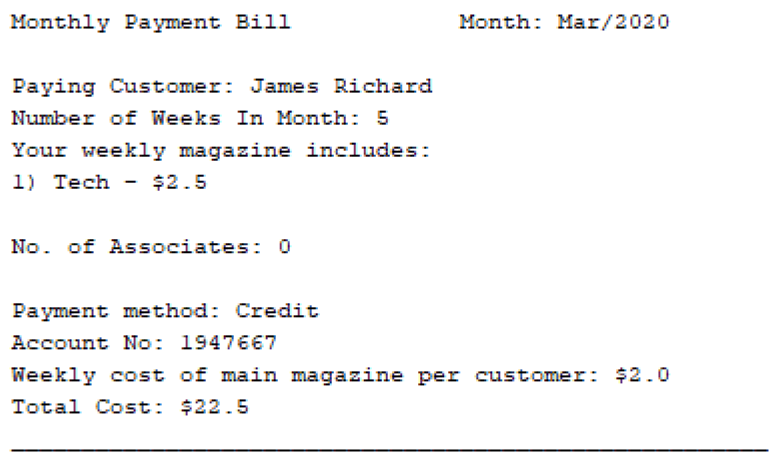
Payment bills after adding associate customer:

(Tom Sawyer)



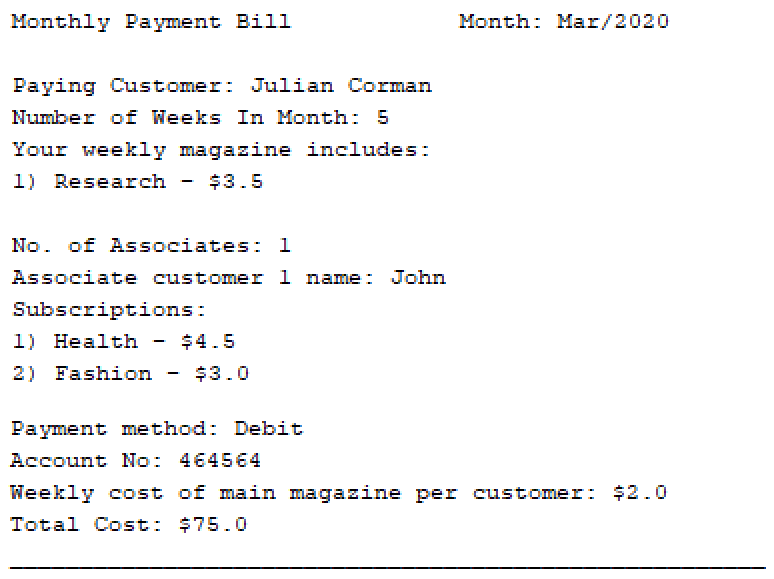
```
Monthly Payment Bill            Month: Mar/2020

Paying Customer: Tom Sawyer
Number of Weeks In Month: 5
Your weekly magazine includes:
1) Tech - $2.5
2) Health - $4.5
3) Beauty - $5.0

No. of Associates: 2
Associate customer 1 name: Alex
Subscriptions:
1) Beauty - $5.0
Associate customer 2 name: Peter
Subscriptions:
1) Beauty - $5.0


Payment method: Debit
Account No: 1234567
Weekly cost of main magazine per customer: $2.0
Total Cost: $140.0
______________________________________________________
```

(James Richard)

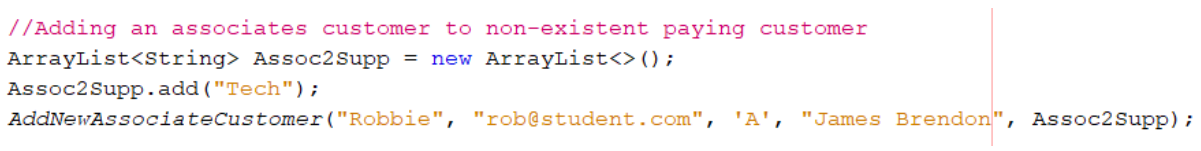


```
Monthly Payment Bill            Month: Mar/2020

Paying Customer: James Richard
Number of Weeks In Month: 5
Your weekly magazine includes:
1) Tech - $2.5

No. of Associates: 0

Payment method: Credit
Account No: 1947667
Weekly cost of main magazine per customer: $2.0
Total Cost: $22.5
______________________________________________________
```

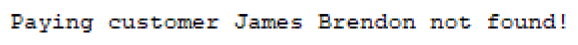
(Julian Corman)


```
Monthly Payment Bill             Month: Mar/2020

Paying Customer: Julian Corman
Number of Weeks In Month: 5
Your weekly magazine includes:
1) Research - $3.5

No. of Associates: 1
Associate customer 1 name: John
Subscriptions:
1) Health - $4.5
2) Fashion - $3.0

Payment method: Debit
Account No: 464564
Weekly cost of main magazine per customer: $2.0
Total Cost: $75.0
______________________________________________________
```

<u>Adding Associate customer to non-existent Paying customer called “James Brendon”:</u>

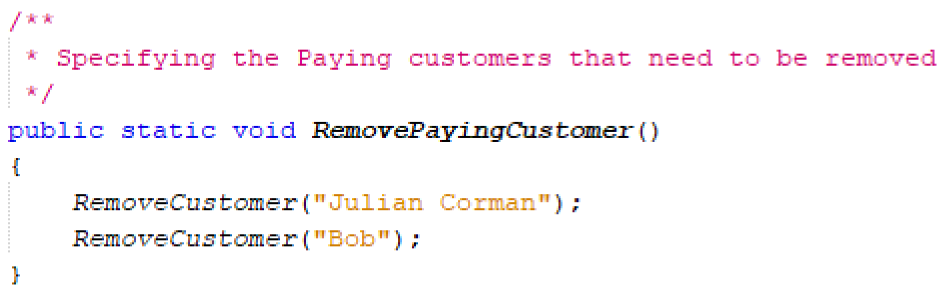

```
//Adding an associates customer to non-existent paying customer
ArrayList<String> Assoc2Supp = new ArrayList<>();
Assoc2Supp.add("Tech");
AddNewAssociateCustomer("Robbie", "rob@student.com", 'A', "James Brendon", Assoc2Supp);
```

<u>A message was displayed saying no such customer exists:</u>


```
Paying customer James Brendon not found!
```

3) Removing Paying customer

<u>Trying to remove 1 paying customer named “Julian Corman” and 1 non-existent customer called “Bob”:</u>


```
/**
 * Specifying the Paying customers that need to be removed
 */
public static void RemovePayingCustomer()
{
    RemoveCustomer("Julian Corman");
    RemoveCustomer("Bob");
}
```

Customer “Julian Corman” was removed successfully, but a message was displayed for customer “Bob” who does not exist:

```
Julian Corman has been removed!

No customer named Bob exists!
```

Viewing the weekly emails after removing customer “Julian Corman”:

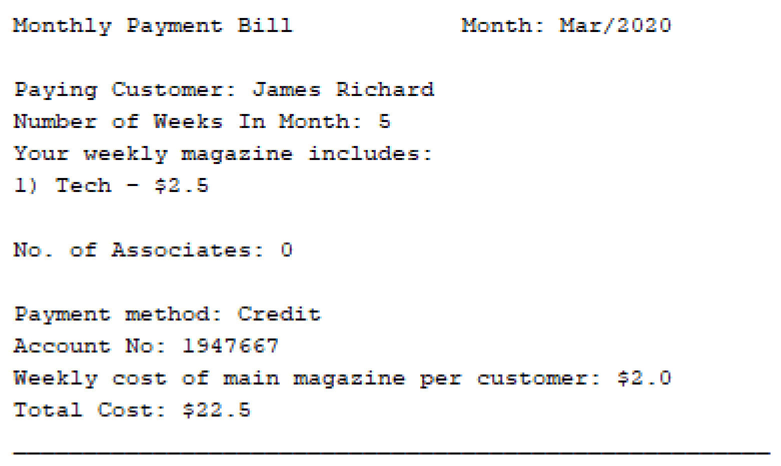
```
Dear Tom Sawyer,                2020-03-24
Your weekly magazine includes:
1) Tech - $2.5
2) Health - $4.5
3) Beauty - $5.0
Associate customer 1 name: Alex
Subscriptions:
1) Beauty - $5.0
Associate customer 2 name: Peter
Subscriptions:
1) Beauty - $5.0


Dear James Richard,             2020-03-24
Your weekly magazine includes:
1) Tech - $2.5
```

Viewing the payment bills after removing paying customer “Julian Corman”:

(Tom Sawyer)

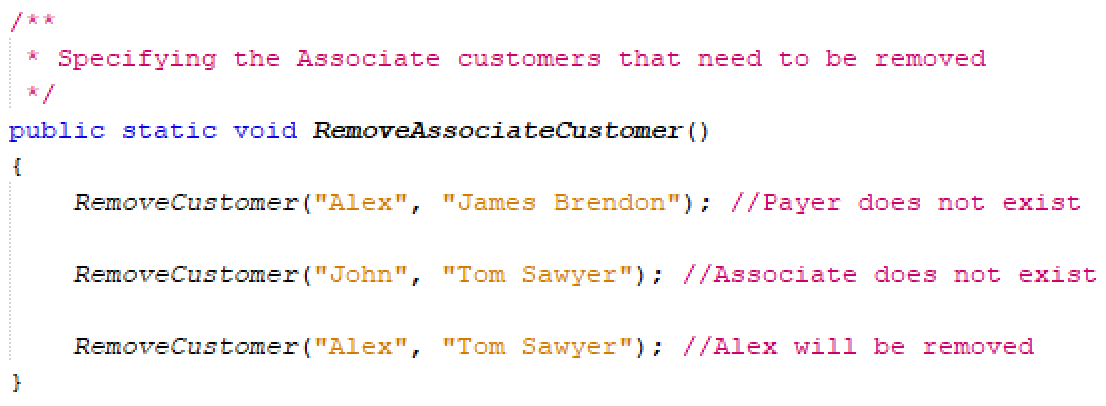
```
Monthly Payment Bill            Month: Mar/2020

Paying Customer: Tom Sawyer
Number of Weeks In Month: 5
Your weekly magazine includes:
1) Tech - $2.5
2) Health - $4.5
3) Beauty - $5.0

No. of Associates: 2
Associate customer 1 name: Alex
Subscriptions:
1) Beauty - $5.0
Associate customer 2 name: Peter
Subscriptions:
1) Beauty - $5.0

Payment method: Debit
Account No: 1234567
Weekly cost of main magazine per customer: $2.0
Total Cost: $140.0
______________________________________________________
```

(James Richard)

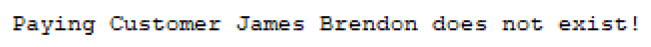

```
Monthly Payment Bill            Month: Mar/2020

Paying Customer: James Richard
Number of Weeks In Month: 5
Your weekly magazine includes:
1) Tech - $2.5

No. of Associates: 0

Payment method: Credit
Account No: 1947667
Weekly cost of main magazine per customer: $2.0
Total Cost: $22.5
______________________________________________________
```

Result: The paying customer “Julian Corman” and all their associate customers were removed from the weekly emails and monthly billing system.

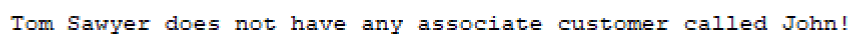
4) Removing Associate customers


```
/**
 * Specifying the Associate customers that need to be removed
 */
public static void RemoveAssociateCustomer()
{
    RemoveCustomer("Alex", "James Brendon"); //Payer does not exist

    RemoveCustomer("John", "Tom Sawyer"); //Associate does not exist

    RemoveCustomer("Alex", "Tom Sawyer"); //Alex will be removed
}
```

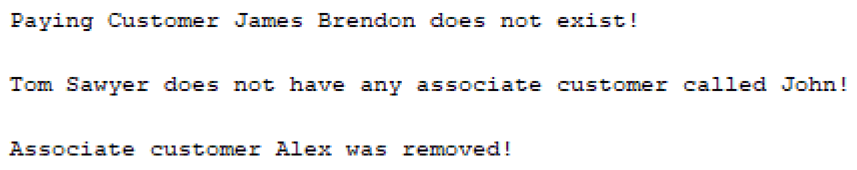
No paying customer named “James Brendon” is present:


```
Paying Customer James Brendon does not exist!
```

Associate customer “John” could not be removed, since, they did not exist:

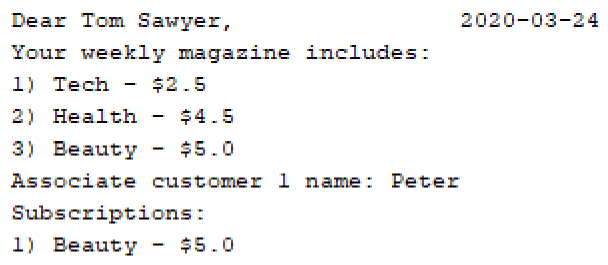

```
Tom Sawyer does not have any associate customer called John!
```

<u>Associate customer “Alex” was removed from paying customer “Tom Sawyer”:</u>



```
Paying Customer James Brendon does not exist!

Tom Sawyer does not have any associate customer called John!

Associate customer Alex was removed!
```

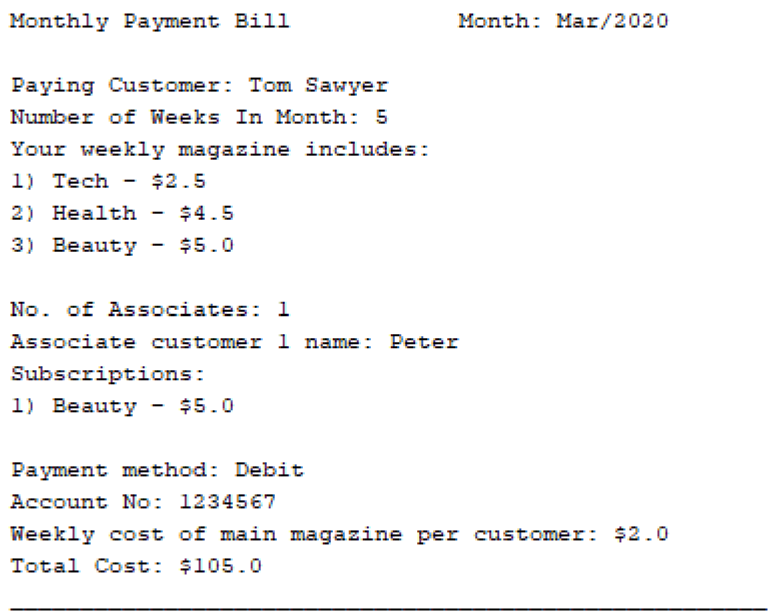
<u>Checking the weekly subscription email for paying customer “Tom Sawyer”:</u>



```
Dear Tom Sawyer,                2020-03-24
Your weekly magazine includes:
1) Tech - $2.5
2) Health - $4.5
3) Beauty - $5.0
Associate customer 1 name: Peter
Subscriptions:
1) Beauty - $5.0
```

<u>Result:</u> Associate customer “Alex” was removed from the mailing system for “Tom Sawyer”.

<u>Checking the monthly payment bill for “Tom Sawyer”:</u>

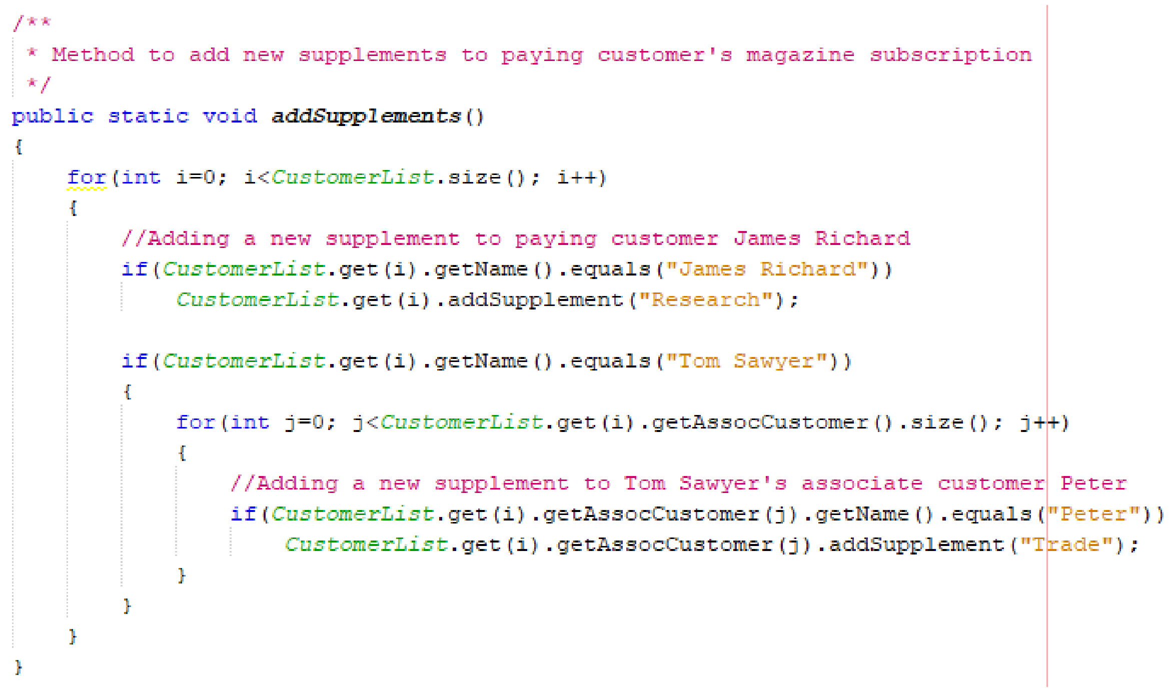


```
Monthly Payment Bill            Month: Mar/2020

Paying Customer: Tom Sawyer
Number of Weeks In Month: 5
Your weekly magazine includes:
1) Tech - $2.5
2) Health - $4.5
3) Beauty - $5.0

No. of Associates: 1
Associate customer 1 name: Peter
Subscriptions:
1) Beauty - $5.0

Payment method: Debit
Account No: 1234567
Weekly cost of main magazine per customer: $2.0
Total Cost: $105.0
_______________________________________________________
```

<u>Result</u>: Associate customer “Alex” was removed from the monthly bill for “Tom Sawyer”.

5) Subscribing to more supplements

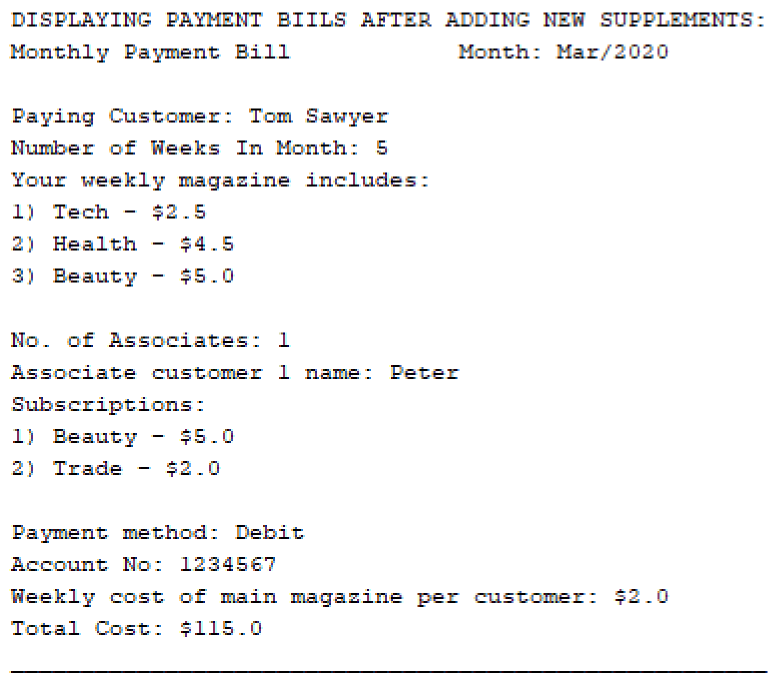
“Research” supplement was added to paying customer “James Richard” and “Trade” supplement to associate customer “Peter”:



```
/**
 * Method to add new supplements to paying customer's magazine subscription
 */
public static void addSupplements()
{
    for(int i=0; i<CustomerList.size(); i++)
    {
        //Adding a new supplement to paying customer James Richard
        if(CustomerList.get(i).getName().equals("James Richard"))
            CustomerList.get(i).addSupplement("Research");

        if(CustomerList.get(i).getName().equals("Tom Sawyer"))
        {
            for(int j=0; j<CustomerList.get(i).getAssocCustomer().size(); j++)
            {
                //Adding a new supplement to Tom Sawyer's associate customer Peter
                if(CustomerList.get(i).getAssocCustomer(j).getName().equals("Peter"))
                    CustomerList.get(i).getAssocCustomer(j).addSupplement("Trade");
            }
        }
    }
}
```

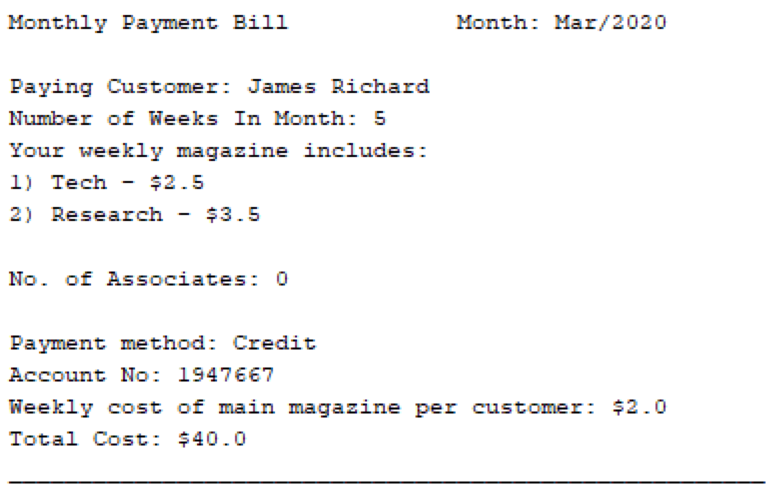
Payment bill after adding the new supplements:

(Tom Sawyer)



```
DISPLAYING PAYMENT BIILS AFTER ADDING NEW SUPPLEMENTS:
Monthly Payment Bill            Month: Mar/2020

Paying Customer: Tom Sawyer
Number of Weeks In Month: 5
Your weekly magazine includes:
1) Tech - $2.5
2) Health - $4.5
3) Beauty - $5.0

No. of Associates: 1
Associate customer 1 name: Peter
Subscriptions:
1) Beauty - $5.0
2) Trade - $2.0

Payment method: Debit
Account No: 1234567
Weekly cost of main magazine per customer: $2.0
Total Cost: $115.0
_______________________________________________________
```

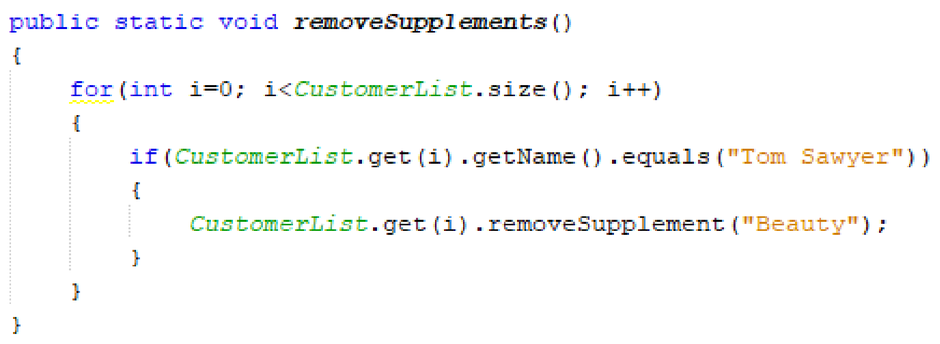
(James Richard)



```
Monthly Payment Bill            Month: Mar/2020

Paying Customer: James Richard
Number of Weeks In Month: 5
Your weekly magazine includes:
1) Tech - $2.5
2) Research - $3.5

No. of Associates: 0

Payment method: Credit
Account No: 1947667
Weekly cost of main magazine per customer: $2.0
Total Cost: $40.0
_____________________________________________________
```

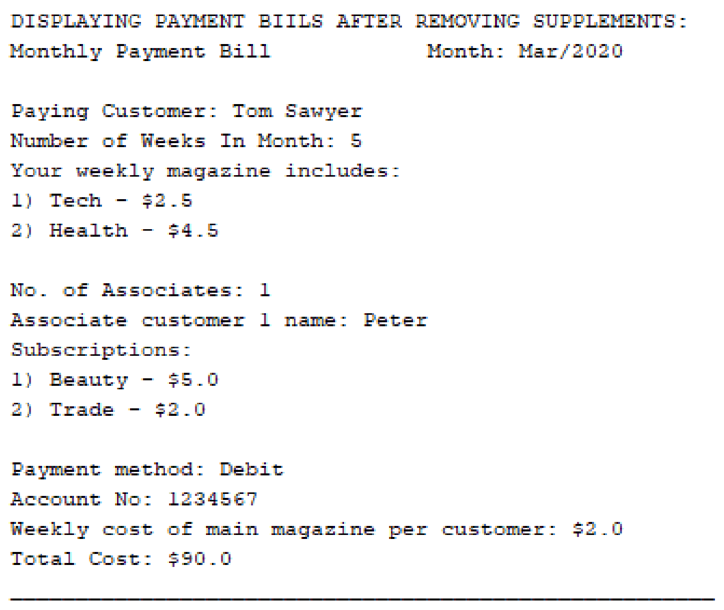
6) Unsubscribing supplements

Removing "Beauty" supplement from paying customer "Tom Sawyer's" magazine subscription:



```java
public static void removeSupplements()
{
    for(int i=0; i<CustomerList.size(); i++)
    {
        if(CustomerList.get(i).getName().equals("Tom Sawyer"))
        {
            CustomerList.get(i).removeSupplement("Beauty");
        }
    }
}
```

Viewing the payment bill removing supplement:



```
DISPLAYING PAYMENT BIILS AFTER REMOVING SUPPLEMENTS:
Monthly Payment Bill            Month: Mar/2020

Paying Customer: Tom Sawyer
Number of Weeks In Month: 5
Your weekly magazine includes:
1) Tech - $2.5
2) Health - $4.5

No. of Associates: 1
Associate customer 1 name: Peter
Subscriptions:
1) Beauty - $5.0
2) Trade - $2.0

Payment method: Debit
Account No: 1234567
Weekly cost of main magazine per customer: $2.0
Total Cost: $90.0
_____________________________________________________
```

Full console output

```
run:
*******************************************
Name: Faiz Syed
Student number: 33243485
Mode of enrolment: internal
Tutor: Mr Ferdous Sohel
Tutorial day and time: Thursday, 10:30 am
*******************************************

ADDING PAYING CUSTOMERS:
No. of paying customer added: 3
Tom Sawyer
James Richard
Julian Corman

ADDING ASSOCIATE CUSTOMERS:
New Associate customer John added to Paying customer Julian Corman


New Associate customer Alex added to Paying customer Tom Sawyer


New Associate customer Peter added to Paying customer Tom Sawyer


Paying customer James Brendon not found!

DISPLAYING SUBSCRIPTION EMAILS:
Dear Tom Sawyer,      2020-03-26
Your weekly magazine includes:
1) Tech - $2.5
2) Health - $4.5
3) Beauty - $5.0
Associate customer 1 name: Alex
Subscriptions:
1) Beauty - $5.0
Associate customer 2 name: Peter
Subscriptions:
1) Beauty - $5.0


Dear James Richard,    2020-03-26
Your weekly magazine includes:
1) Tech - $2.5


Dear Julian Corman,    2020-03-26
Your weekly magazine includes:
1) Research - $3.5
Associate customer 1 name: John
```

```
Subscriptions:
1) Health - $4.5
2) Fashion - $3.0


DISPLAYING PAYMENT BILLS:
Monthly Payment Bill       Month: Mar/2020

Paying Customer: Tom Sawyer
Number of Weeks In Month: 5
Your weekly magazine includes:
1) Tech - $2.5
2) Health - $4.5
3) Beauty - $5.0

No. of Associates: 2
Associate customer 1 name: Alex
Subscriptions:
1) Beauty - $5.0
Associate customer 2 name: Peter
Subscriptions:
1) Beauty - $5.0

Payment method: Debit
Account No: 1234567
Weekly cost of main magazine per customer: $2.0
Total Cost: $140.0
____________________________________________________


Monthly Payment Bill       Month: Mar/2020

Paying Customer: James Richard
Number of Weeks In Month: 5
Your weekly magazine includes:
1) Tech - $2.5

No. of Associates: 0

Payment method: Credit
Account No: 1947667
Weekly cost of main magazine per customer: $2.0
Total Cost: $22.5
____________________________________________________


Monthly Payment Bill       Month: Mar/2020

Paying Customer: Julian Corman
Number of Weeks In Month: 5
Your weekly magazine includes:
```

```
1) Research - $3.5

No. of Associates: 1
Associate customer 1 name: John
Subscriptions:
1) Health - $4.5
2) Fashion - $3.0

Payment method: Debit
Account No: 464564
Weekly cost of main magazine per customer: $2.0
Total Cost: $75.0
____________________________________________________________


REMOVING PAYING CUSTOMERS:
Julian Corman has been removed!

No customer named Bob exists!

REMOVING ASSOCIATE CUSTOMERS:
Paying Customer James Brendon does not exist!

Tom Sawyer does not have any associate customer called John!

Associate customer Alex was removed!

DISPLAYING SUBSCRIPTION EMAILS AFTER REMOVING CUSTOMERS:
Dear Tom Sawyer,      2020-03-26
Your weekly magazine includes:
1) Tech - $2.5
2) Health - $4.5
3) Beauty - $5.0
Associate customer 1 name: Peter
Subscriptions:
1) Beauty - $5.0


Dear James Richard,    2020-03-26
Your weekly magazine includes:
1) Tech - $2.5


DISPLAYING PAYMENT BIILS AFTER REMOVING CUSTOMERS:
Monthly Payment Bill      Month: Mar/2020

Paying Customer: Tom Sawyer
Number of Weeks In Month: 5
Your weekly magazine includes:
1) Tech - $2.5
2) Health - $4.5
```

```
3) Beauty - $5.0

No. of Associates: 1
Associate customer 1 name: Peter
Subscriptions:
1) Beauty - $5.0

Payment method: Debit
Account No: 1234567
Weekly cost of main magazine per customer: $2.0
Total Cost: $105.0
_____________________________________________________


Monthly Payment Bill     Month: Mar/2020

Paying Customer: James Richard
Number of Weeks In Month: 5
Your weekly magazine includes:
1) Tech - $2.5

No. of Associates: 0

Payment method: Credit
Account No: 1947667
Weekly cost of main magazine per customer: $2.0
Total Cost: $22.5
_____________________________________________________


DISPLAYING PAYMENT BIILS AFTER ADDING NEW SUPPLEMENTS:
Monthly Payment Bill     Month: Mar/2020

Paying Customer: Tom Sawyer
Number of Weeks In Month: 5
Your weekly magazine includes:
1) Tech - $2.5
2) Health - $4.5
3) Beauty - $5.0

No. of Associates: 1
Associate customer 1 name: Peter
Subscriptions:
1) Beauty - $5.0
2) Trade - $2.0

Payment method: Debit
Account No: 1234567
Weekly cost of main magazine per customer: $2.0
Total Cost: $115.0
_____________________________________________________
```

```
Monthly Payment Bill      Month: Mar/2020

Paying Customer: James Richard
Number of Weeks In Month: 5
Your weekly magazine includes:
1) Tech - $2.5
2) Research - $3.5

No. of Associates: 0

Payment method: Credit
Account No: 1947667
Weekly cost of main magazine per customer: $2.0
Total Cost: $40.0
____________________________________________________


DISPLAYING PAYMENT BIILS AFTER REMOVING SUPPLEMENTS:
Monthly Payment Bill      Month: Mar/2020

Paying Customer: Tom Sawyer
Number of Weeks In Month: 5
Your weekly magazine includes:
1) Tech - $2.5
2) Health - $4.5

No. of Associates: 1
Associate customer 1 name: Peter
Subscriptions:
1) Beauty - $5.0
2) Trade - $2.0

Payment method: Debit
Account No: 1234567
Weekly cost of main magazine per customer: $2.0
Total Cost: $90.0
____________________________________________________


Monthly Payment Bill      Month: Mar/2020

Paying Customer: James Richard
Number of Weeks In Month: 5
Your weekly magazine includes:
1) Tech - $2.5
2) Research - $3.5

No. of Associates: 0
```

```
Payment method: Credit
Account No: 1947667
Weekly cost of main magazine per customer: $2.0
Total Cost: $40.0
____________________________________________________


BUILD SUCCESSFUL (total time: 0 seconds)
```

Test Table:

No.	Test	Expected result	Actual result	Pass/ Fail?
1	Adding 3 paying customers	3 paying customers should be added	3 paying customers were added	Pass
2	View weekly subscription emails after adding paying customers	Subscription emails should be displayed	Subscription emails were displayed	Pass
3	View monthly payment bill after adding paying customers	Payment bills should be displayed	Payment bills were displayed	Pass
4	Adding Associate customers to existing paying customers “Julian Corman” and “Tom Sawyer”	New associate customers should be added to the corresponding paying customers	New associate customers “John”, “Alex” and “Peter” were added	Pass
5	Adding associate customers to non-existent paying customer called “James Brendon”	A messaging saying “James Brendon” does not exist should be displayed	A message saying, “Paying Customer James Brendon does not exist!” was presented	Pass
6	View subscription emails after adding associate customers	New updated subscription emails showing the newly added associate customers should be displayed	New updated subscription emails showing the newly added associate customers was displayed	Pass
7	View payment bills after adding associate customers	New updated payment bills including the newly added associate customers should be displayed	New updated payment bills including the newly added associate customers was displayed	Pass
8	Removing an existing paying customer “Julian Corman”	The given paying customer should be removed	The given paying customer was removed	Pass
9	Removing a non-existent paying customer “Bob”	A message should be displayed	A message saying, “No customer named Bob exists!” was displayed	Pass

10	View subscription email after removing the requested paying customers	The specified paying customer and all its associate customers should be removed from the email system	The specified paying customer and all its associate customers were removed from the email system	Pass
11	View payment bill after removing the requested paying customers	The specified paying customer and all its associate customers should be removed from the payment bill system	The specified paying customer and all its associate customers were removed from the payment bill system	Pass
12	Removing an associate customer from a non-existent paying customer called "James Brendon"	A message should be displayed	A message was displayed saying, "Paying customer James Brendon does not exist!"	Pass
14	Removing an associate customer "John" that does not belong to the given paying customer "Tom Sawyer"	A message should be displayed	A message saying, "Tom Sawyer does not have any associate customer called John!" was displayed	Pass
15	Removing an associate customer from the right paying customer	The associate customer should be removed	The given associate customer was removed from the given paying customer	Pass
16	View subscription email after removing associate customers	Associate customers should be removed from the subscription email system	Associate customers were also removed from the subscription email system	Pass
17	View payment bill after removing associate customers	Associate customers should be removed from the payment bill for the paying customers	Associate customers were removed from the payment bill	Pass
18	Add new supplement to the magazine subscription	New supplement should get added	New supplement gets added	Pass
19	View payment bill after adding supplements	The payment bill should be increased	The new supplement cost was added to the payment bill	Pass
20	Remove supplements from magazine subscription	The customer should be able to unsubscribe supplements	The customer was able to unsubscribe the specified supplement	Pass
21	View payment bill after removing a supplement	The payment bill should be adjusted accordingly	The unsubscribed supplement cost was reduced from the payment bill	Pass
22	Trying to remove customer from empty ArrayList	A message should be displayed	A message indicating an empty list is displayed	Pass

Resources used:

https://docs.oracle.com/javase/8/docs/api/java/util/ArrayList.html

https://howtodoinjava.com/java/date-time/java-date-examples/

https://www.youtube.com/watch?v=OOdO785p3Qo

https://www.roseindia.net/answers/viewqa/Java-Beginners/12012-How-to-calculate-number-of-weeks-in-a-specified-month.html

https://stackoverflow.com/questions/4338267/validate-phone-number-with-javascript

https://stackoverflow.com/questions/3605214/javascript-add-leading-zeroes-to-date

https://stackoverflow.com/questions/7388001/javascript-regex-to-validate-date-format

https://www.w3schools.com/jsref/event_onchange.asp

https://www.w3schools.com/jsref/event_onclick.asp

https://www.w3schools.com/tags/tag_select.asp

https://www.javatpoint.com/array-vs-arraylist-in-java