CS1003 W03 Practical 170025298 13 Feb 2018

Overview:

This practical mainly aims at testing and exercise read and write from .csv file and learn to use BufferedReader and PrintWriter method, which can also be concluded as try to read and write from files. This report figure out the process of programming W03- Practical and the the design of this program. Testing and example of the program runs will also be contained. At the end of the report the evaluation will be conclude and evalued.

Note have been made properly in the program.

Design:

This program contained 3 classes. They are:

Main class

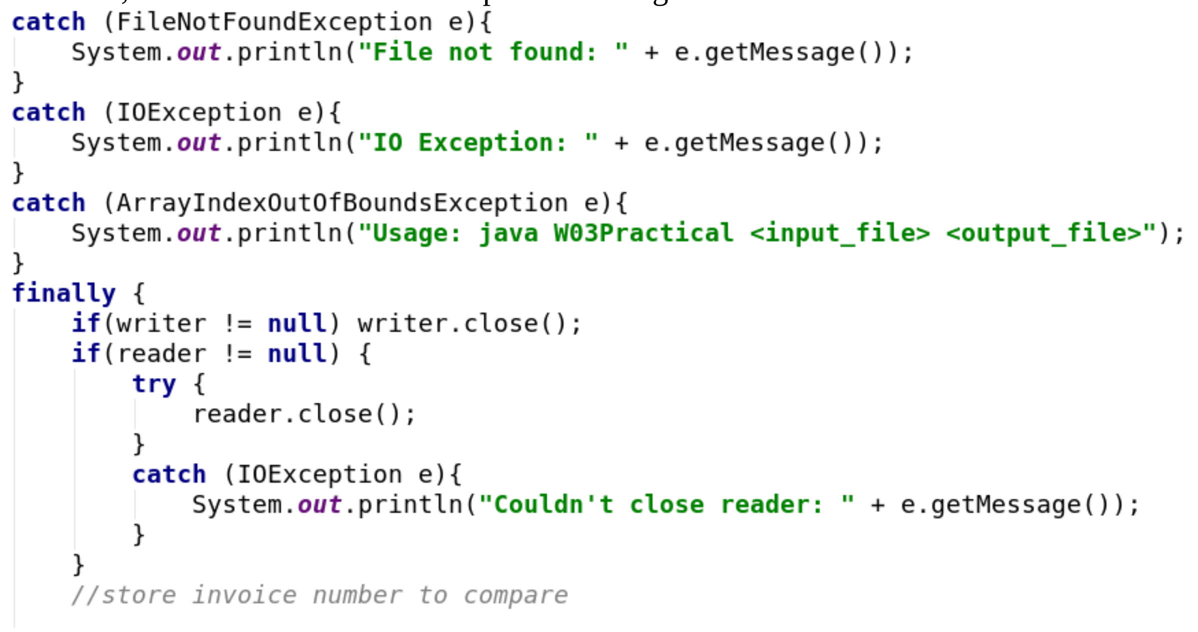
Read and write class(RW in program)

protector(For security)

In the main class, only the direction to rw class is contained. RW class contains 6 methods, which are:

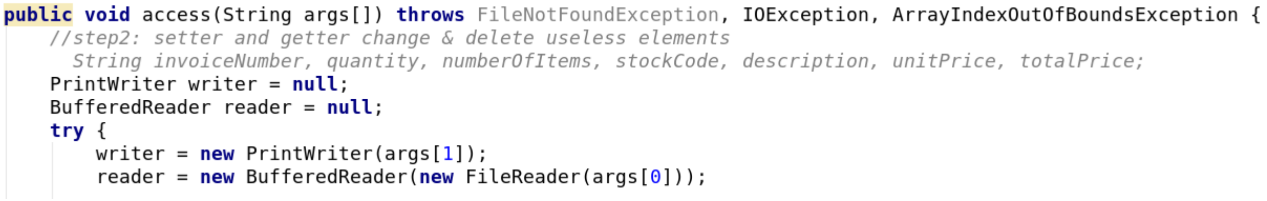
Access – void:

This method is the most important part of the program, which is used to access to the csv file and re-write the data into .txt file. Try and catch are used in this program to throw FileNotFoundExcption, IOexception and ArrayIndexOutOfBoundException. In try and catch command, these has been wrote the replaced message :



```
catch (FileNotFoundException e){
    System.out.println("File not found: " + e.getMessage());
}
catch (IOException e){
    System.out.println("IO Exception: " + e.getMessage());
}
catch (ArrayIndexOutOfBoundsException e){
    System.out.println("Usage: java W03Practical <input_file> <output_file>");
}
finally {
    if(writer != null) writer.close();
    if(reader != null) {
        try {
            reader.close();
        }
        catch (IOException e){
            System.out.println("Couldn't close reader: " + e.getMessage());
        }
    }
    //store invoice number to compare
```

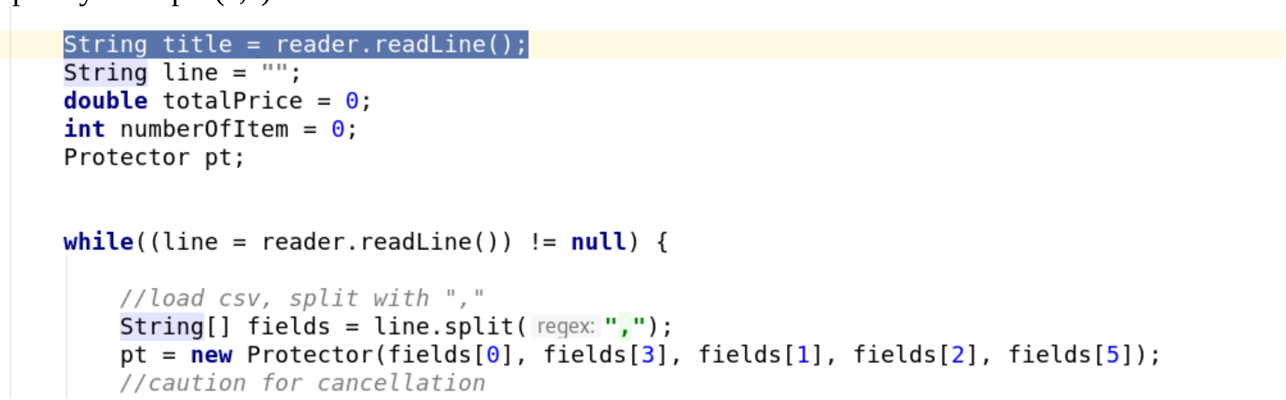
Decimal format package have been imported before class and this class is used to control the decimal number into two. Args[] has been write to the parameter of this method to fit to the main class.



```
public void access(String args[]) throws FileNotFoundException, IOException, ArrayIndexOutOfBoundsException {
    //step2: setter and getter change & delete useless elements
      String invoiceNumber, quantity, numberOfItems, stockCode, description, unitPrice, totalPrice;
    PrintWriter writer = null;
    BufferedReader reader = null;
    try {
        writer = new PrintWriter(args[1]);
        reader = new BufferedReader(new FileReader(args[0]));
```

Two argument is organised in this way.

The title of .csv file is only be read but not wrote to the document by readline() the title. Data is split by line.split(“,”) and the detail is at the follow:



```
String title = reader.readLine();
String line = "";
double totalPrice = 0;
int numberOfItem = 0;
Protector pt;


while((line = reader.readLine()) != null) {

    //load csv, split with ","
    String[] fields = line.split( regex: ",");
    pt = new Protector(fields[0], fields[3], fields[1], fields[2], fields[5]);
    //caution for cancellation
```

compare-boolean

Compare method is for compare whether two methods are the same.

pre_InvoiceNumber

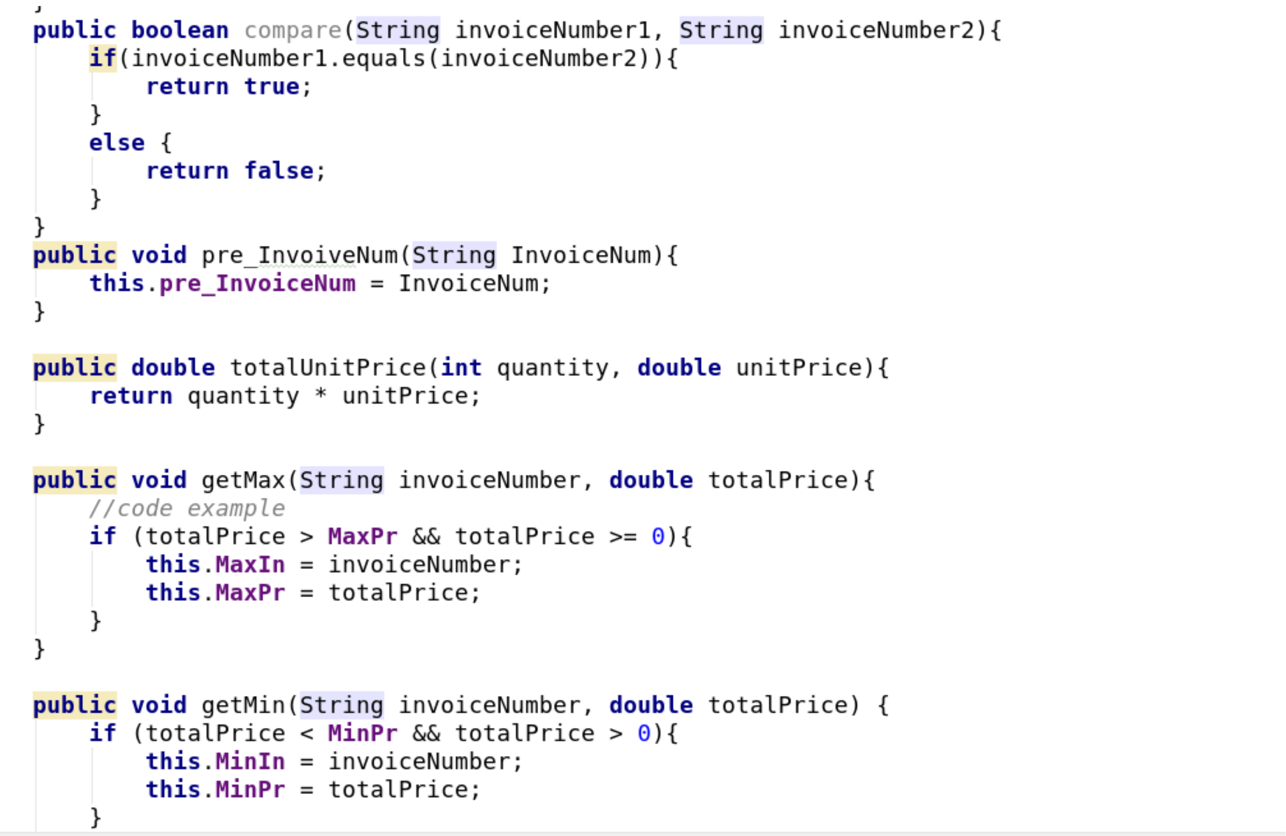
This method is to store the previous invoice number

totalUnitPrice-double

This method is to calculate the total unit price

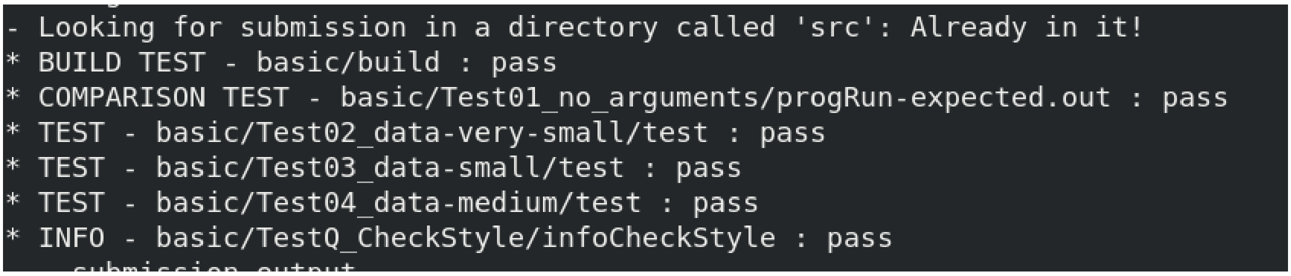
getMax/getMin-void

This method is use to get the main data from the csv via compare



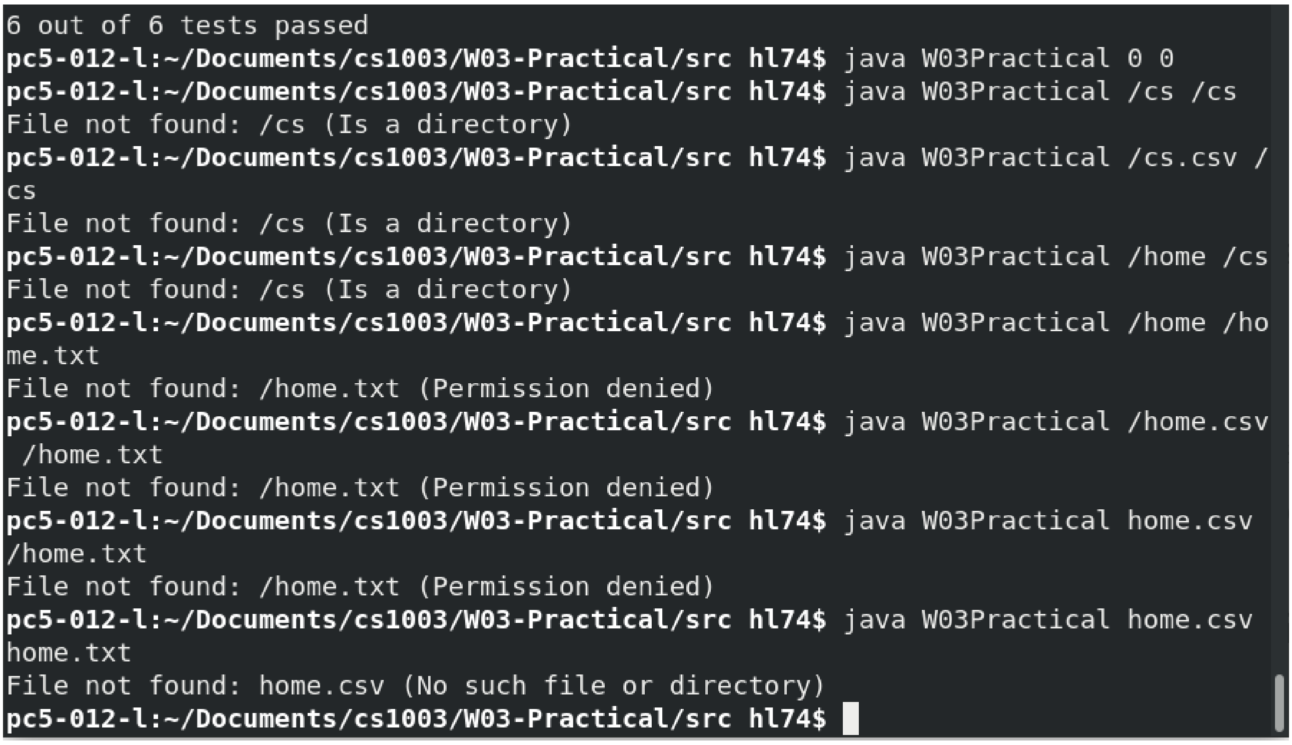
```
public boolean compare(String invoiceNumber1, String invoiceNumber2){
    if(invoiceNumber1.equals(invoiceNumber2)){
        return true;
    }
    else {
        return false;
    }
}
public void pre_InvoiveNum(String InvoiceNum){
    this.pre_InvoiceNum = InvoiceNum;
}

public double totalUnitPrice(int quantity, double unitPrice){
    return quantity * unitPrice;
}

public void getMax(String invoiceNumber, double totalPrice){
    //code example
    if (totalPrice > MaxPr && totalPrice >= 0){
        this.MaxIn = invoiceNumber;
        this.MaxPr = totalPrice;
    }
}

public void getMin(String invoiceNumber, double totalPrice) {
    if (totalPrice < MinPr && totalPrice > 0){
        this.MinIn = invoiceNumber;
        this.MinPr = totalPrice;
    }
```

Testing:
Stacscheck:



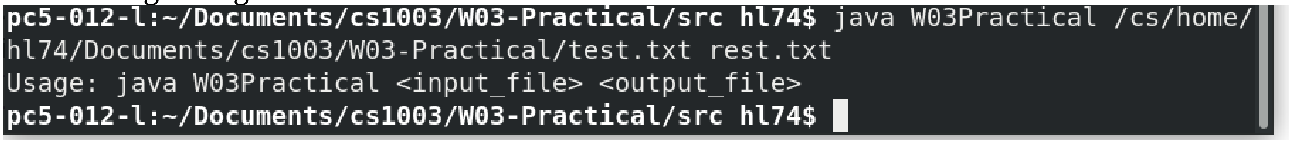
```
- Looking for submission in a directory called 'src': Already in it!
* BUILD TEST - basic/build : pass
* COMPARISON TEST - basic/Test01_no_arguments/progRun-expected.out : pass
* TEST - basic/Test02_data-very-small/test : pass
* TEST - basic/Test03_data-small/test : pass
* TEST - basic/Test04_data-medium/test : pass
* INFO - basic/TestQ_CheckStyle/infoCheckStyle : pass
- submission output
```

while file not found:



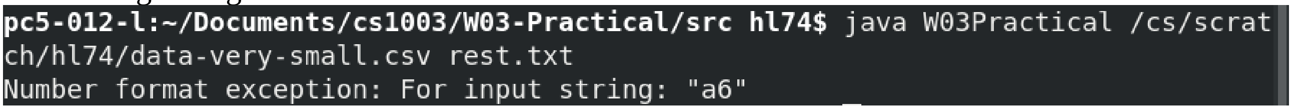
```
6 out of 6 tests passed
pc5-012-l:~/Documents/cs1003/W03-Practical/src hl74$ java W03Practical 0 0
pc5-012-l:~/Documents/cs1003/W03-Practical/src hl74$ java W03Practical /cs /cs
File not found: /cs (Is a directory)
pc5-012-l:~/Documents/cs1003/W03-Practical/src hl74$ java W03Practical /cs.csv /
cs
File not found: /cs (Is a directory)
pc5-012-l:~/Documents/cs1003/W03-Practical/src hl74$ java W03Practical /home /cs
File not found: /cs (Is a directory)
pc5-012-l:~/Documents/cs1003/W03-Practical/src hl74$ java W03Practical /home /ho
me.txt
File not found: /home.txt (Permission denied)
pc5-012-l:~/Documents/cs1003/W03-Practical/src hl74$ java W03Practical /home.csv
 /home.txt
File not found: /home.txt (Permission denied)
pc5-012-l:~/Documents/cs1003/W03-Practical/src hl74$ java W03Practical home.csv
/home.txt
File not found: /home.txt (Permission denied)
pc5-012-l:~/Documents/cs1003/W03-Practical/src hl74$ java W03Practical home.csv
home.txt
File not found: home.csv (No such file or directory)
pc5-012-l:~/Documents/cs1003/W03-Practical/src hl74$
```

While using wrong format file:



```
pc5-012-l:~/Documents/cs1003/W03-Practical/src hl74$ java W03Practical /cs/home/
hl74/Documents/cs1003/W03-Practical/test.txt rest.txt
Usage: java W03Practical <input_file> <output_file>
pc5-012-l:~/Documents/cs1003/W03-Practical/src hl74$
```

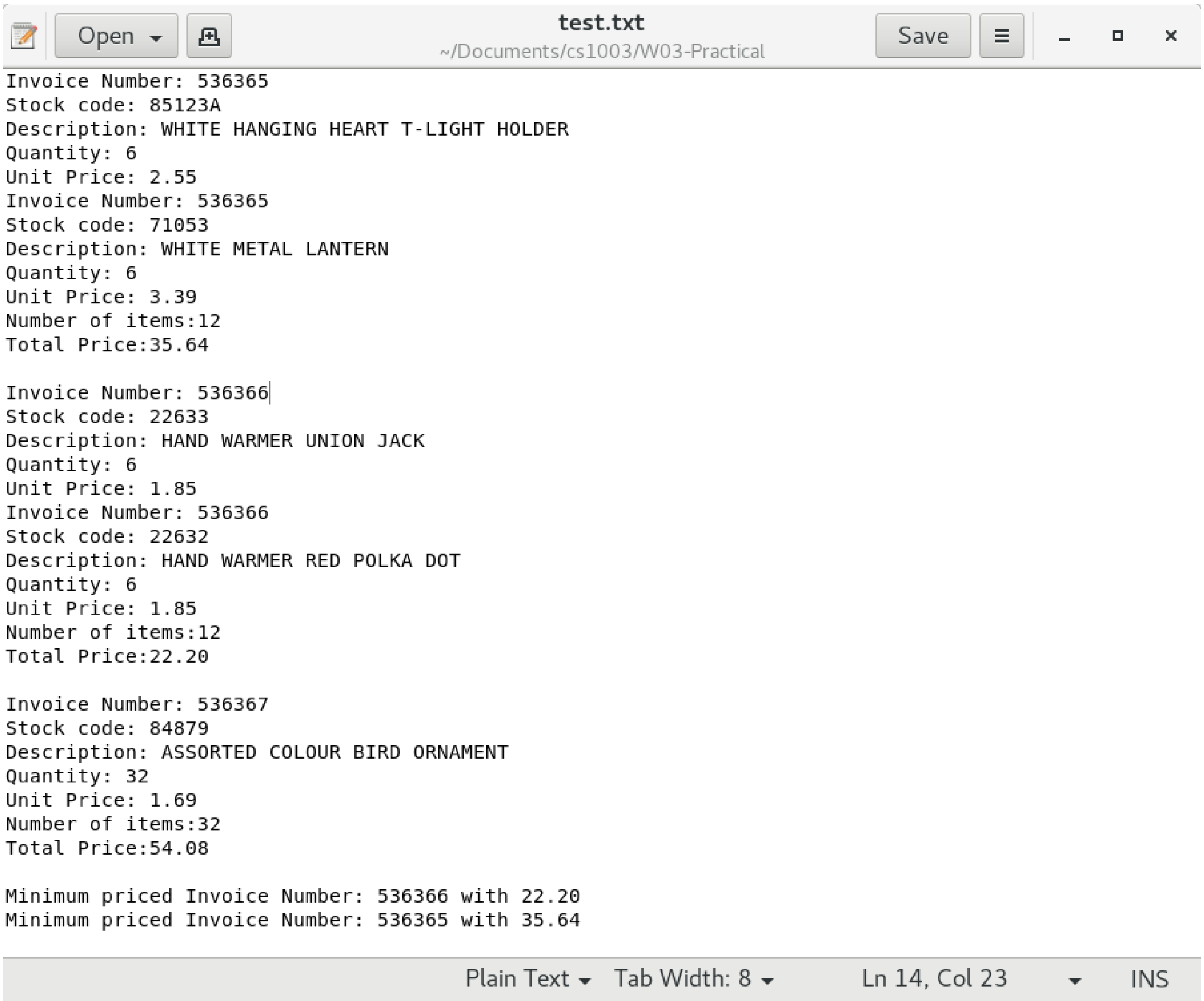
while using wrong fomat data:



```
pc5-012-l:~/Documents/cs1003/W03-Practical/src hl74$ java W03Practical /cs/scrat
ch/hl74/data-very-small.csv rest.txt
Number format exception: For input string: "a6"
```

Test end.

Examples:



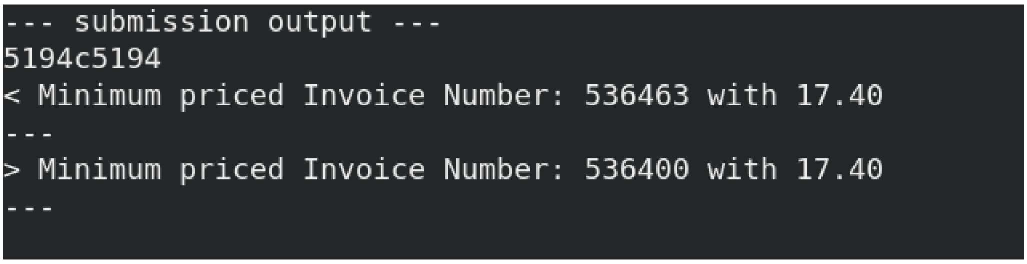
```
Invoice Number: 536365
Stock code: 85123A
Description: WHITE HANGING HEART T-LIGHT HOLDER
Quantity: 6
Unit Price: 2.55
Invoice Number: 536365
Stock code: 71053
Description: WHITE METAL LANTERN
Quantity: 6
Unit Price: 3.39
Number of items:12
Total Price:35.64

Invoice Number: 536366
Stock code: 22633
Description: HAND WARMER UNION JACK
Quantity: 6
Unit Price: 1.85
Invoice Number: 536366
Stock code: 22632
Description: HAND WARMER RED POLKA DOT
Quantity: 6
Unit Price: 1.85
Number of items:12
Total Price:22.20

Invoice Number: 536367
Stock code: 84879
Description: ASSORTED COLOUR BIRD ORNAMENT
Quantity: 32
Unit Price: 1.69
Number of items:32
Total Price:54.08

Minimum priced Invoice Number: 536366 with 22.20
Minimum priced Invoice Number: 536365 with 35.64
```

This is the screen shot from data_very_small.csv the program runs well and good to use.

Evaluation:
To make the program be more secured, the data of the total number and total price should might be saved in protector to keep the data safe.
List command might be used to make the data be read and save and change and finally write.
From stacscheck, @ should be use to make the java comment and I will repair the formative issue as much as possible before hand in.

Conclusion:
The program is well organized and works well. Structures of the code is also good organised.
New knowledges have been used properly and knowledge from last semester has also been reviewed. Comments has been used well in program and method has been used properly. As for this result, it is detailed and every steps has been finished, there might be some grammar mistake but after all, My report should be good enough. Extension will be wrote in report soon.

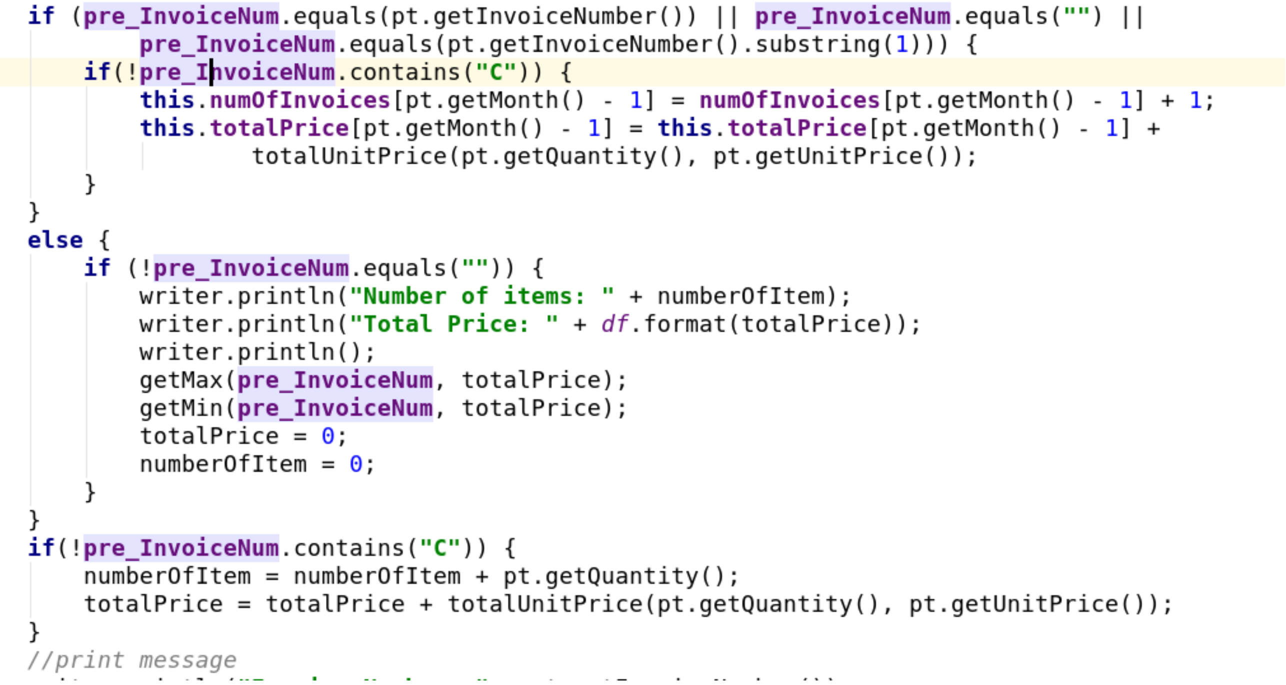
Tips: Invoice Number of 536463 and 536400 have same total price from stacscheck in data-small.csv



```
--- submission output ---
5194c5194
< Minimum priced Invoice Number: 536463 with 17.40
---
> Minimum priced Invoice Number: 536400 with 17.40
---
```

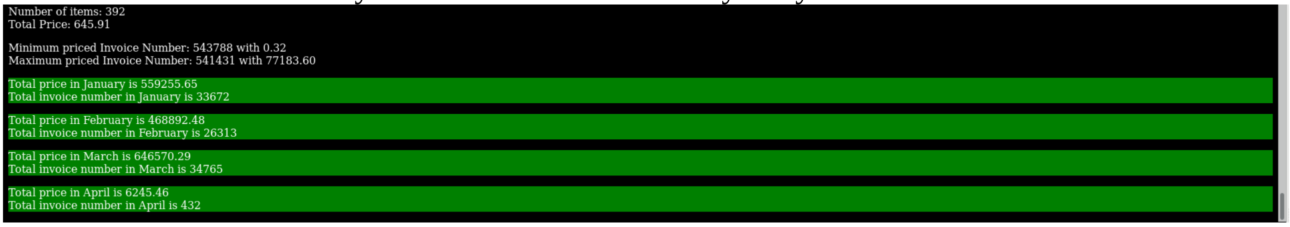
Extension:

1. cancellation: All of the invoice number started with C, C have been deleted and this will not be concluded in both number of invoices(Extension 2) and the number and total price will be 0 in this trade.

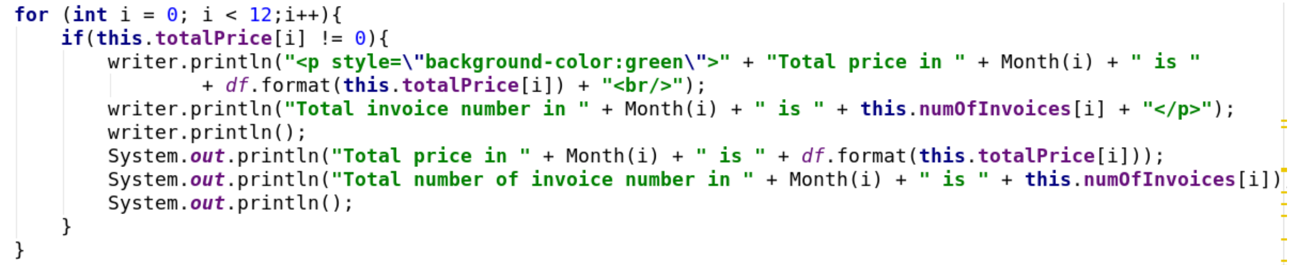


```
if (pre_InvoiceNum.equals(pt.getInvoiceNumber()) || pre_InvoiceNum.equals("") ||
        pre_InvoiceNum.equals(pt.getInvoiceNumber().substring(1))) {
    if(!pre_InvoiceNum.contains("C")) {
        this.numOfInvoices[pt.getMonth() - 1] = numOfInvoices[pt.getMonth() - 1] + 1;
        this.totalPrice[pt.getMonth() - 1] = this.totalPrice[pt.getMonth() - 1] +
                totalUnitPrice(pt.getQuantity(), pt.getUnitPrice());
    }
}
else {
    if (!pre_InvoiceNum.equals("")) {
        writer.println("Number of items: " + numberOfItem);
        writer.println("Total Price: " + df.format(totalPrice));
        writer.println();
        getMax(pre_InvoiceNum, totalPrice);
        getMin(pre_InvoiceNum, totalPrice);
        totalPrice = 0;
        numberOfItem = 0;
    }
}
if(!pre_InvoiceNum.contains("C")) {
    numberOfItem = numberOfItem + pt.getQuantity();
    totalPrice = totalPrice + totalUnitPrice(pt.getQuantity(), pt.getUnitPrice());
}
//print message
```

2. I screen shotted the picture from webpage and the data is from data-medium2.csv. The program is available to divide data by month which is achieved by array:

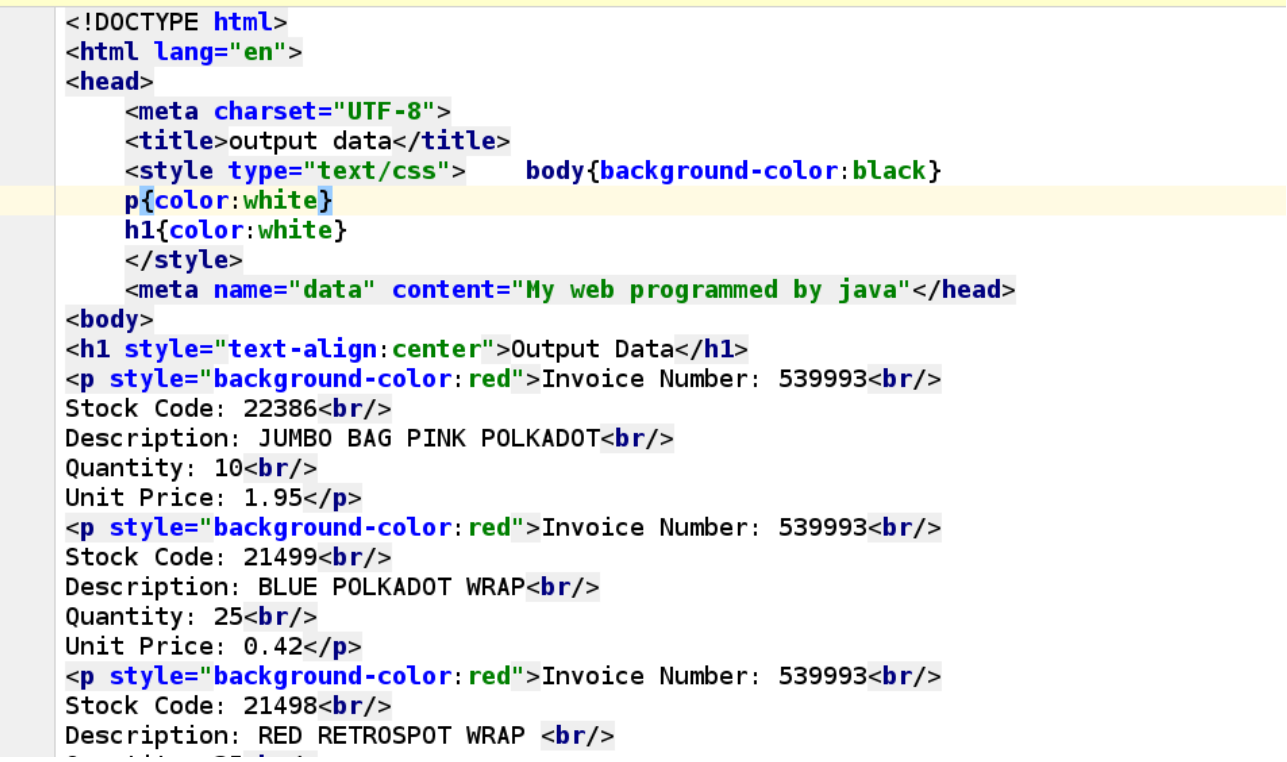


```
Number of items: 392
Total Price: 645.91

Minimum priced Invoice Number: 543788 with 0.32
Maximum priced Invoice Number: 541431 with 77183.60

Total price in January is 559255.65
Total invoice number in January is 33672

Total price in February is 468892.48
Total invoice number in February is 26313

Total price in March is 646570.29
Total invoice number in March is 34765

Total price in April is 6245.46
Total invoice number in April is 432
```

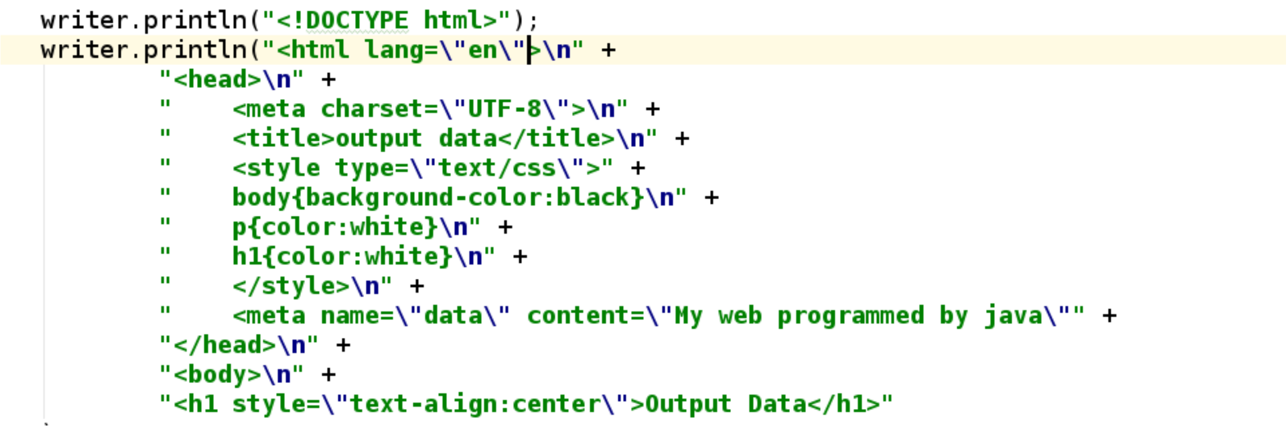


```
for (int i = 0; i < 12;i++){
    if(this.totalPrice[i] != 0){
        writer.println("<p style=\"background-color:green\">" + "Total price in " + Month(i) + " is "
                + df.format(this.totalPrice[i]) + "<br/>");
        writer.println("Total invoice number in " + Month(i) + " is " + this.numOfInvoices[i] + "</p>");
        writer.println();
        System.out.println("Total price in " + Month(i) + " is " + df.format(this.totalPrice[i]));
        System.out.println("Total number of invoice number in " + Month(i) + " is " + this.numOfInvoices[i])
        System.out.println();
    }
}
```

3. I use writer to write the html and I can't really tell how do I do this, but the detailed program is like followed:

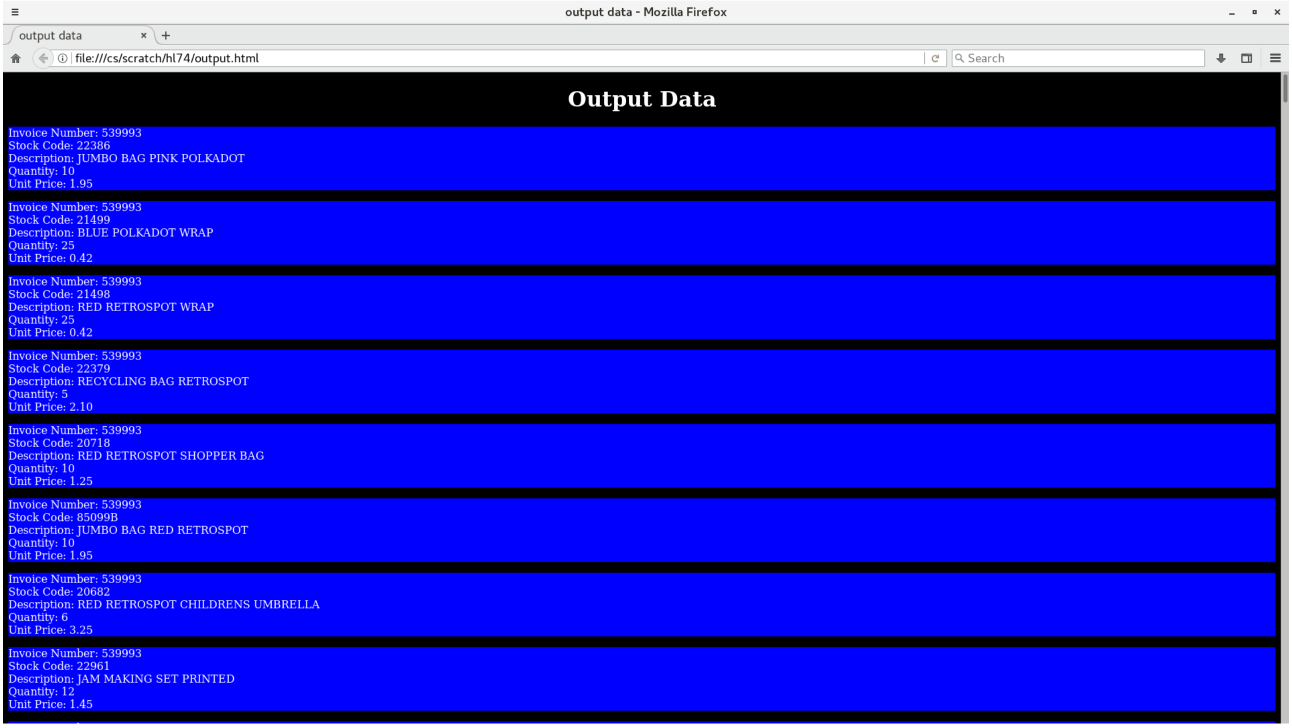


```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>output data</title>
    <style type="text/css">    body{background-color:black}
    p{color:white}
    h1{color:white}
    </style>
    <meta name="data" content="My web programmed by java"</head>
<body>
<h1 style="text-align:center">Output Data</h1>
<p style="background-color:red">Invoice Number: 539993<br/>
Stock Code: 22386<br/>
Description: JUMBO BAG PINK POLKADOT<br/>
Quantity: 10<br/>
Unit Price: 1.95</p>
<p style="background-color:red">Invoice Number: 539993<br/>
Stock Code: 21499<br/>
Description: BLUE POLKADOT WRAP<br/>
Quantity: 25<br/>
Unit Price: 0.42</p>
<p style="background-color:red">Invoice Number: 539993<br/>
Stock Code: 21498<br/>
Description: RED RETROSPOT WRAP <br/>
```



```
writer.println("<!DOCTYPE html>");
writer.println("<html lang=\"en\">\n" +
        "<head>\n" +
        "    <meta charset=\"UTF-8\">\n" +
        "    <title>output data</title>\n" +
        "    <style type=\"text/css\">" +
        "    body{background-color:black}\n" +
        "    p{color:white}\n" +
        "    h1{color:white}\n" +
        "    </style>\n" +
        "    <meta name=\"data\" content=\"My web programmed by java\"" +
        "</head>\n" +
        "<body>\n" +
        "<h1 style=\"text-align:center\">Output Data</h1>"
```

I use print writer to write the code into html file and create a new html code and the appearance is like follows:


output data - Mozilla Firefox
output data
file:///cs/scratch/hl74/output.html
Search

```
Output Data

Invoice Number: 539993
Stock Code: 22386
Description: JUMBO BAG PINK POLKADOT
Quantity: 10
Unit Price: 1.95

Invoice Number: 539993
Stock Code: 21499
Description: BLUE POLKADOT WRAP
Quantity: 25
Unit Price: 0.42

Invoice Number: 539993
Stock Code: 21498
Description: RED RETROSPOT WRAP
Quantity: 25
Unit Price: 0.42

Invoice Number: 539993
Stock Code: 22379
Description: RECYCLING BAG RETROSPOT
Quantity: 5
Unit Price: 2.10

Invoice Number: 539993
Stock Code: 20718
Description: RED RETROSPOT SHOPPER BAG
Quantity: 10
Unit Price: 1.25

Invoice Number: 539993
Stock Code: 85099B
Description: JUMBO BAG RED RETROSPOT
Quantity: 10
Unit Price: 1.95

Invoice Number: 539993
Stock Code: 20682
Description: RED RETROSPOT CHILDRENS UMBRELLA
Quantity: 6
Unit Price: 3.25

Invoice Number: 539993
Stock Code: 22961
Description: JAM MAKING SET PRINTED
Quantity: 12
Unit Price: 1.45
```

These are all extensions I write and Thank you for looking at my report.