Hacettepe University
Department of Computer Engineering
BBM104 Introduction to Programming Laboratory II
Programming Assignment 1

Submission Date	**: 25.02.2016**
Due Date	**: 09.03.2016**
Advisor	**: Dr. Fuat AKAL, R.A. Alaettin UÇAN**
Programing Language	**: JAVA**
Title	**: Pharmacy Stock Management**

INTRODUCTION

In this experiment you are expected to gain knowledge on basic JAVA programming. The program you are going to develop will deal with variables, loops, string operations and file read operations. Besides the programming task, you will also learn to comply with coding standards and to document your program by using Javadoc.

PROBLEM

In this experiment, you are expected to implement a Pharmacy Stock Management System. You will read prescriptions and price list from input files. For reading data from input files, you can use the code given in Appendix A. Your task is to calculate total cost of a prescription based on the price list. While developing your program, you are also supposed document it. To achieve this, you will use Javadoc (See Appendix B for some hints).

1. Prescription File (Input)

Prescriptions will be available as files in the format below:

[patient name and surname] *tab* **[social security administration]** *tab* **[prescription date]** *newline*

[medicament name] *tab* **[quantity]** *newline*

[medicament name] *tab* **[quantity]** *newline*

...

According to the prescription file format, the prescription file contains the **patient's name, prescription date** and **related social security administration code** in the first line. Social security administration code can be either ES (*Emekli Sandığı*), SSK (*Sosyal Sigortalar Kurumu*) or BK (*Bağ-Kur*). Medicaments prescribed are listed in the following lines. There may be at least one or more medicaments for every prescription. Lines for the medicaments in the file contains the **name of the medicine** and its **quantity**. Every item in the file separated with a **tab** character. A sample prescription file is shown in Figure 1**.**

Mehmet Uzun		SSK	26.01.2016
Aspirin	1		
Novalgin	2		

Figure 1 Prescription Example

2. Price List File (Input)

Prices list for medicaments for their validity periods are also kept in an input file. This file has the format of:

[medicament name] *tab* **[social security institution]** *tab* **[validity date]** *tab* **[expiry date]** *tab* **[price]** *newline*

[medicament name] *tab* **[social security institution]** *tab* **[validity date]** *tab* **[expiry date]** *tab* **[price]** *newline*

[medicament name] *tab* **[social security institution]** *tab* **[validity date]** *tab* **[expiry date]** *tab* **[price]** *newline*

...

Every line in the file has the **medicament name, social security administration code, effective date, expiry date** and **price** (in Turkish Lira) columns separated by a tab character. Same medicament can appear in the file multiple times. You should pay attention to the social security administration codes and validity dates of medicaments. Same medicament can occur in the file several times with different prices. An example file is shown in Figure 2**.**

Aspirin	SSK	01.01.2016	31.01.2016	5,5
Aspirin	SSK	01.02.2016	29.02.2016	6,2
Aspirin	ES	01.01.2016	31.01.2016	5,4
Aspirin	BK	01.01.2016	31.01.2016	6,1
Aspirin	SSK	01.01.2016	31.01.2016	5,4
Novalgin	SSK	01.01.2016	31.01.2016	8,3

Figure 2 Medicaments Price List Example

3. Finding Right Price

For every prescribed medicine, there is one right price. The medicine is looked up in the price list in the order of name, social security administration code and validity dates. If there is more than one item is found, the one with the minimum price is the right price for that medicine.

4. Invoicing the Prescription

Once the entire prescription is processed and the right prices are calculated for every medicine in the prescription, an invoice is generated to reflect the total cost. The invoice will be in the following format:

[medicament name] *tab* **[unit price]** *tab* **[quantity]** *tab* **[amount]** *newline* Total **[Total amount]**

An example invoice for the prescription given in Figure 1 is seen in Figure 3.

Aspirin	5,4	1	5,4
Novalgin	8,3	2	16,6
Total 22,00			

Figure 3 Invoice Example

Submit Format

File hierarchy must be zipped before submitted (Not .rar, only .zip files are supported by the system)
<student id>.zip
- javadoc.zip (See : https://en.wikipedia.org/wiki/Javadoc)
- src.zip (Main.java, *.java)
- Report.pdf

Late Policy

You may use up to three extension days for the assignment. But each extension day will bring about additional 10% degradation for evaluation of the assignment.

Notes and Restrictions

- Every method and class must be documented using Javadoc.
- Save all your work until the assignment is graded.
- All assignments must be original, individual work. Duplicate or very similar assignments are both going to be considered as cheating.

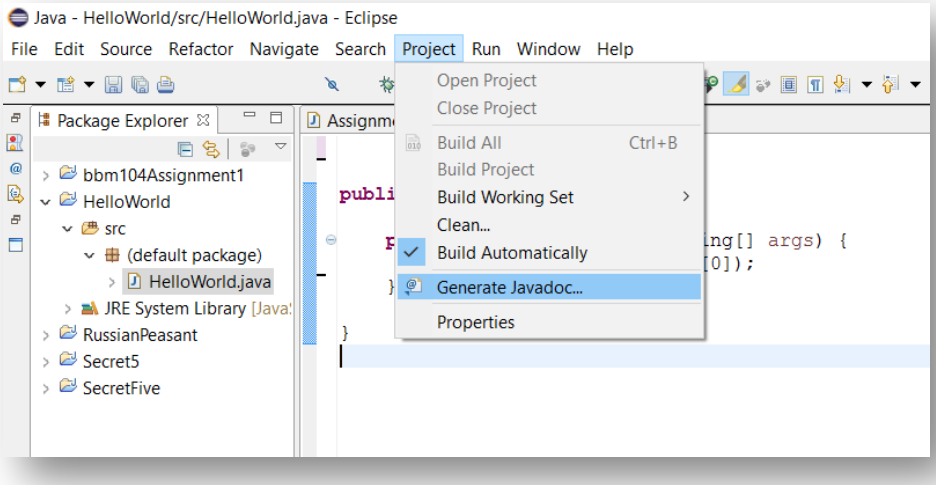
Appendix A.

You can use the following code to read your input files.

```
import java.io.IOException;
import java.nio.file.Files;
import java.nio.file.Paths;

public class ReadFromFile {

    public static String[] readFile(String path) {
        try {
        int i =0;
        int length= Files.readAllLines(Paths.get(path)).size();
        String[] results = new String[length];
            for (String line : Files.readAllLines(Paths.get(path))) {
                results[i++]= line;
            }
        return results;
        } catch (IOException e) {
            e.printStackTrace();
            return null;
        }
    }

    public static void main(String[] args) {
        String[] lines= readFile("testfile.txt");
        for (String line : lines) {
            System.out.println(line);
            }
    }
}
```

Appendix B.

JavaDoc is a documentation generator for generating API documentation in HTML format from Java source code. The HTML format is used to add the convenience of being able to hyperlink related documents together. The "doc comments" format used by Javadoc is the defacto industry standard for documenting Java classes. Eclipse, automatically generate Javadoc HTML.



Structure of a Javadoc comment

A Javadoc comment is set off from code by standard multi-line comment tags /* and */. The opening tag (called begin-comment delimiter), has an extra asterisk, as in /**.

The first paragraph is a description of the method documented.

Following the description are a varying number of descriptive tags, signifying:

- The parameters of the method (@param)
- What the method returns (@return)
- Any exceptions the method may throw (@throws)
- Other less-common tags such as @see (a "see also" tag) [1]

```
// import statements

/**
 * @author      Firstname Lastname <address @ example.com>
 * @version     1.6                (current version number of program)
 * @since       2010-03-31         (the version of the package this class was first added to)
 */
public class Test {
    // class body
}
```

References

[1] https://en.wikipedia.org/wiki/Javadoc