First Assignment –
Objected Oriented Programming

Matheus Moretti Rangel – 21250414

Contents:

Introduction

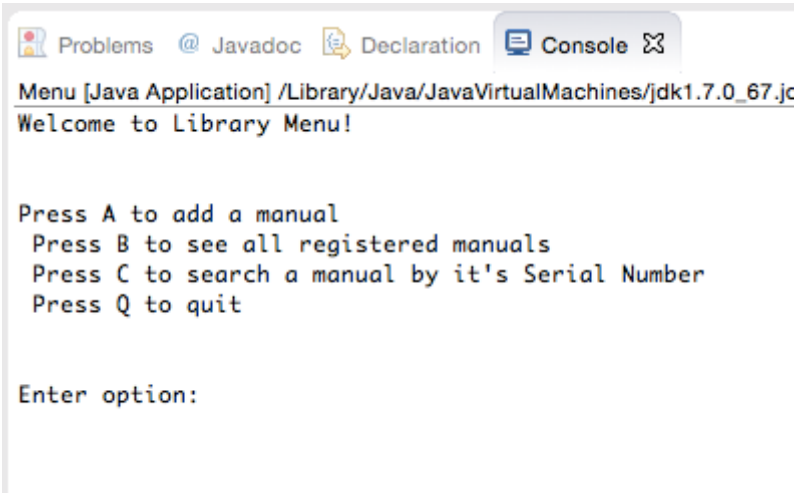
The goal of this assignment was to make an console-based application to help the “IT Centre of a small company” to register and keep record of it’s technical manuals. This application was written in Java, using the Eclipse IDE. It was not required to export a JAR (Java Archive) file nor send the Java classes. All pieces of code will be seen on this documentation later on.

The application basically prompts a menu to the user, asking him whether he/she wants to add a new manual, see all registered manuals, search a manual by it’s serial number (an unique ID) or quit the application. This menu will be running on a loop until the user quit the application.

User Guide

By simply running the classes via Eclipse, pressing the above button,

the main function of the Menu class will start running on your console. This message will appear:



Enter option A by simply typing this character and pressing enter. A first message asking the manual's Serial Number will appear. Type it in and press enter. Repeat this process for the manual's Title and Author.

```
Enter option: A
Enter manual's Serial Number:   123456
Enter manual's Title:   Title
Enter manual's Author:   Author
```

By entering option B, a list with all registered manuals will be shown. You could see, for example, the manual we just recorded.

```
Enter option: B
Serial Number:  123456  Title:  Title  Author:  Author
```

If option C is chosen, you must input a Serial Number to be searched. If there is already registry for such Serial Number, the following will be shown:

```
Enter option: C
Please input the Serial Number:   123456
Serial Number:  123456  Title:  Title  Author:  Author
```

Otherwise:

```
Enter option: c
Please input the Serial Number:   2
No manual under such Serial Number
Welcome to Library Menu!
```

By pressing the Quit option, the program stops running:

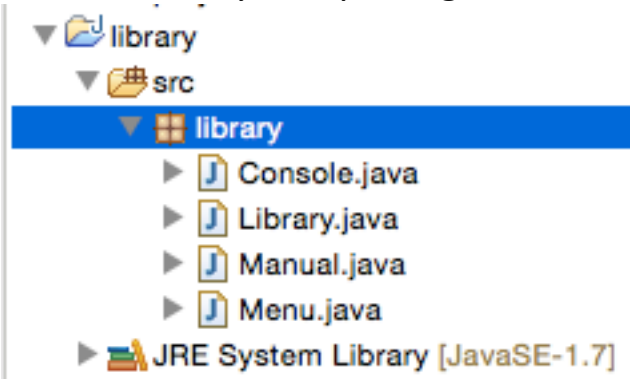
```
Enter option: q
Thank you for using the app.
```

If any other option is input, the following will appear:

```
Enter option: y
I'm afraid this is not a valid option.
Welcome to Library Menu!
```

Documentation

Following the example provided structure, all 4 classes are inside only one package, called library.



Menu.java – This is the main class. It is responsible for printing the Menu on the console. Inside it's main function, a library object will be created. That means every time we run the main class our Library is a brand-new object, with no records inside. For this assignment no saving files were used. We have a loop printing the menu and waiting for user input. It was also a viable and more-optimized option to create 2 functions and print this menu on a recursive way. I chose not to do it once I followed the structures and comments of the similar example shown to us.

The user input is treated inside a switch:

- The add option will inside the menu class ask for the Serial Number. I have, then, an "if" condition that analyses the existence of such serial number, to avoid duplicated register. If no manual under that Serial is found, we can create a Manual object, asking (manual class ask method) it's details and adding (library class addManual method) it to a manual's arraylist and getting (manual getSerialNumber method) it's Serial Number to add to an arraylist with all serial numbers. This list was made to make it quicker the searches. It's not needed to

run through all manual details if you want to check the presence of a Serial Number.

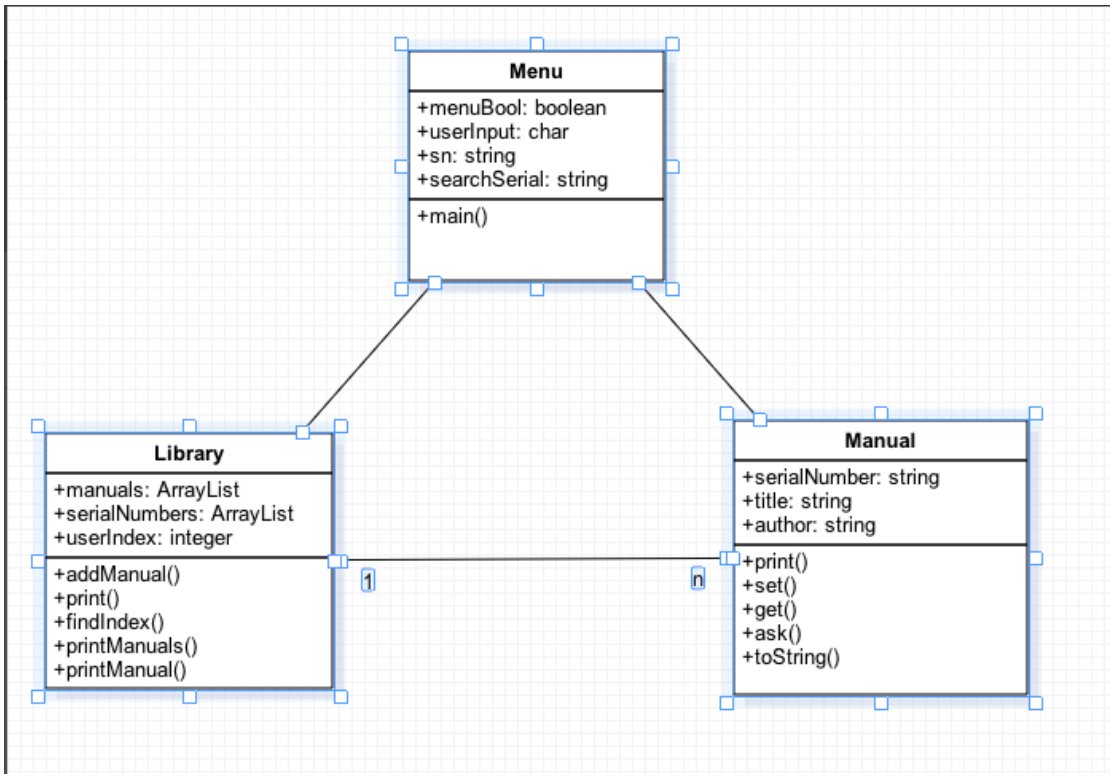
- The B option calls our library printManuals method. It is basically a for loop from 0 until the end of our manuals arraylist, printing each reference (after making it a string).
- The C option will ask for a Serial Number and call the printManual method on our Library class. This printManual will look for an existence of such Serial on our serials arraylist. If there is any registry, we will print it. Otherwise, we inform the user there is no manual under this reference.
- The Q option will change our Boolean variable and make the loop terminates.
- Any other input will enter the Default condition and inform the user that is not a valid option.

Manual.java – After declaring our three variables (author, title and serialNumber) we have a default constructor that will set those values to default values. The same constructor method can be called with parameters and set those parameters as values. To do such thing, we call a set method. I chose to do 3 get methods although only 1 was used. They will simply return a value. We have a print method, which was only used on previous testing. I decided not to erase it once it could be used on later versions. Our ask method will define values to title and author variables, once the serialNumber value was defined on the menu class (as mentioned above). Our toString method overrides the homonymous method and makes it easier for our user to read the information.

Library.java – After creating arraylists for manuals and one exclusively to our serial numbers, we can see the addManual method, responsible for appending to each one of those the manual values and the serial number values. Our findIndex

method will return us the position of some serial number inside our arraylist. All other methods were explained above

Class Diagram



Printed Listing of Classes

Library.java | Menu.java | Manual.java | Console.java

```
 */
package library;
public class Menu {
    public static void main(String[] args) {
        // Initialize container
        Library library = new Library();
        // declare boolean for while loop
        boolean menuBool = true;
        // while loop
        while (menuBool) {
            // ask option using menu choices
            System.out.println("Welcome to Library Menu!");
            System.out.println("\n\nPress A to add a manual"
                    + "\n Press B to see all registered manuals"
                    + "\n Press C to search a manual by it's Serial Number"
                    + "\n Press Q to quit");
            // implement switch option
            char userInput = Console.askChar("\n");
            // askChar will return us an upper case character
            switch (userInput) {
            case 'A':
            // adding manuals. I check if findIndex function returns any position. If it does not,
                // we create a new manual.
                String sn = Console.askString("Enter manual's Serial Number:   ");
                if (library.findIndex(sn) >= 0) {
                    System.out.println("This Serial is already registered.");
                }
                else {
                    Manual newManual = new Manual();
                    newManual.ask(sn);
                    library.addManual(newManual);
                }
                break;
            // print
            case 'B':
                library.printManuals();
                break;
            // find
            case 'C':
                String searchSerial = Console.askString("Please input the Serial Number:   ");
                library.printManual(searchSerial);
                break;
            // quit menu case
            case 'Q':
                System.out.println("Thank you for using the app.");
                menuBool = false;
                break;

            // default case
            default:
                System.out.println("I'm afraid this is not a valid option.");
            }
```

menu.java

Library.java | Menu.java | *Manual.java | Console.java

```java
/*
 * Matheus Moretti Rangel - 21250414
 * Assignment 1 - OOP
 */
package library;

public class Manual {
        // declare variables
        String author;
        String title;
        String serialNumber;
        // default constructor
        public Manual () {
            set("??????", "Untitled", "Unknown");
        }
        // constructor with parameters
        public Manual(String serialNumber, String title, String author) {
            set(serialNumber, title, author);
        }
        // set method
        public void set(String serialNumber, String title, String author) {
            this.serialNumber = serialNumber;
            this.title = title;
            this.author = author;
        }
        // get method
        public  String getTitle (){
            return title;
        }
        public  String getAuthor (){
            return author;
        }
        public  String getSerialNumber (){
            return serialNumber;
        }

        // print method
        public void print() {
            System.out.println("Serial Number: " +serialNumber);
            System.out.println("Title: " +title);
            System.out.println("Author: " +author);
        }
        // ask method for user input
        public void ask(String sn) {
            String title = Console.askString("Enter manual's Title:   ");
            String author = Console.askString("Enter manual's Author:   ");
            set(sn, title, author);
        }
        // toString method
        public String toString() {
            return "Serial Number:  " + serialNumber + "  Title:  " + title + "  Author:  " + author;
        }
}
```

manual.java

Library.java | Menu.java | Manual.java | Console.java

```java
/*
 * Matheus Moretti Rangel - 21250414
 * Assignment 1 - OOP
 */
package library;

import java.util.ArrayList;

public class Library
{
    // declare arraylists
    ArrayList<Manual> manuals;
    ArrayList<String> serialNumbers;

    // construct arraylist for manuals and Serial Numbers
    public Library () {
    manuals = new ArrayList<Manual>();
    serialNumbers = new ArrayList<String>();
    }

    // add manual
    public void addManual (Manual manual) {
    manuals.add(manual);
    serialNumbers.add(manual.getSerialNumber());
    }
    // find index for specified serialNumber
    public int findIndex (String serialNumber) {
        return serialNumbers.indexOf(serialNumber);
    }

    // print manuals
    public void printManuals () {
        for (int i = 0; i < manuals.size(); i++ ) {
            System.out.println((manuals.get(i)).toString());
        }
    }

    public void printManual(String inputSerialNumber) {
        int userIndex = findIndex(inputSerialNumber);
        if (userIndex >= 0) {
            System.out.println((manuals.get(userIndex).toString()));
        }
        else {
            System.out.println("No manual under such Serial Number");
        }
    }
}
```

library.java

Testing Details

I chose not to create test classes mainly to make sure I would have a testing scenario very similar to the final use of this application. I could not find any bugs or program misbehaviours.