****

**Cooperative Management**

**System for University of Makati**

in partial fulfillment

of the courses’ requirements in

**Computer Programming 1**

1st Semester, Academic Year 2019-2020

Presented to

Prof. Jomariss Plan

Presented by

Sherwyne R. Costiniano

1 - ACDS

October 26, 2019

**OBJECTIVES OF THE PROJECT**

The main objective of the project is to design and implement a cooperative management system that will ease the procedures and transactions done manually both for the members and management of the society. In this case, the society are the faculty members, professors, and students.

**The objective of the project includes:**

* + To eliminate inconveniency and repetitive process generally suffered by the management and members of the co-operative society.
  + To ensure data security and controlled user access to data by limiting their access respectively.
  + To ensure accuracy and speed in terms of transactions made.
  + To ease processing and retrieving data.

**PROJECT DESCRIPTION**

Cooperative Management System for University of Makati is a management system that will ensure to ease repetitive and tiring process and transactions made by every members of the society. To ensure data security it comes up with Registration and Login Form to limit the access to a specified user. It has 3 different user access - these are the Student, Professor, and Admin.

In the Admin side, it has features like CRUD (Create, Remove, Update, Delete) to be able to manage the users and the resources. To make the program easy to read, the data of the resources and users will be displayed in table that can be arranged accordingly.

In the Professor side, it has features like Loan System which consists of loaning a money, paying the loan, and checking the current balance of the loan. The professor can also get a book without any payment so that the professor can have a reference to what will he/she discuss.

And lastly, in the Student side, it has features like buying an item which already have a pre-defined items to select like Books, Lace, Scantron, and Uniform. It also have a feature to display the purchase history of the student in which the student can see the item bought, quantity, price, total payment, amount paid by the student, and change. The system also generates a unique receipt code that can be used for an another feature which is Refund/Return of Item.

**CONCEPTS AND TOPICS APPLIED**

1. Data Structures and Data types
   1. Arrays
   2. ArrayList
   3. List
   4. HashMap
   5. Boolean
   6. String
   7. Double
   8. Int
   9. Object
2. Object Oriented Programming
   1. Inheritance
   2. Encapsulation
3. Functions and Classes
   1. Method Overriding
4. Control Structures / Conditional Statements
   1. Nested if – else if – else
   2. Switch statement
   3. Inline if else statement eg: SOP(condition ? True : False);
5. Looping
   1. Nested for loop
   2. While loop
6. Javax Swing Libraries
   1. JOptionPane
   2. JTable
   3. JPasswordField
   4. JScrollPane
7. Algorithms
   1. Search Algorithm
   2. Login Authentication and Validation
   3. Data Sanitation and Input Validation
   4. CRUD (Create, Read, Update, Delete) Algorithm

**SAMPLE TEST DATA**

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case #** | **Action** | **Input** | **Output** |
| 1 | Register as a Student | Name: Sherwyne  User Type: Student  Password: sherwyne | Registered Successfully!  Your UserID is:  STUDENT 1 |
| 2 | Register as a Professor | Name: Dr. Sherwyne  User Type: Professor  Password: drsherwyne | Registered Successfully!  Your UserID is: PROFESSOR 1 |
| 3 | Enter nothing in all input | N/A | Invalid Input / User Not Found / Item Not Found |
| 4 | Login as Student using recently registered student | UserID: STUDENT1  Password: sherwyne | Go to Main Menu of Student with Welcome Message and Options |
| 5 | Login as Professor using recently registered professor | UserID: PROFESSOR1  Password: drsherwyne | Go to Main Menu of Professor with Welcome Message and Options |
| 6 | Enter non-existing account in Student Login | UserID: FAKE  Password: fake | Student Not Found! |
| 7 | Enter non-existing account in Professor Login | UserID: FAKE  Password: fake | Professor Not Found |
| 8 | Enter existing account in Professor Login but with wrong password | UserID: PROFESSOR1  Password: wrong | Incorrect UserID or Password! |
| 9 | Enter existing account in Student Login but with wrong password | UserID: STUDENT1  Password: wrong | Incorrect UserID or Password! |
| 10 | Login as Admin | UserID: admin  Password: admin | Go to Main Menu of Admin with Welcome Message and Options |
| 11 | Add a User in Admin Menu  (Manage Users > Add User)  Note: Login the User after adding to verify the action | User Type: Student or Professor  User ID: STUDENT2  Name: Costiniano  Password: costiniano | Prompts a confirm option that display that Student Info. Then if YES, it will display Successfully Added! |
| 12 | View the previously added User in Admin Menu  (Manage Users > View User) | Student ID: STUDENT2 | STUDENT INFORMATION  Student Name: Costiniano  Password: admin |
| 13 | Update the previously added User in Admin Menu  (Manage Users > Update User)  Note: Login the User after updating to verify the action | Student ID: STUDENT2  New Name: Rosete  New Password: rosete | Prompts a confirm option that display the new info. Then if YES, it will display Successfully Updated! |
| 14 | Delete the previously added User in Admin Menu  (Manage Users > Delete User) | STUDENT ID: STUDENT2 | Prompts a confirm option that displays the user to be delete. Then if YES, it will display Successfully Deleted! |
| 15 | Login as Professor then make a Loan, Check the balance, Pay the Loan then Check the balance again | Amount Loan: 9999  Amount Pay Loan: 999 | In the first Check Balance, the amount will be 9999, then in the second Check Balance will be 9000 since you paid only 999. |
| 16 | Login as Student then buy an item, and check the Purchase History  (Items > Buy > Book | Item Choice: Book  Book ID: book1  Quantity: 1  Amount to Pay: 150 | You will get a change of 49.00php and get a receipt code called STUDENT1book1  Then in Purchase History you will see the Item you bought recently |

**PROGRAM LISTING**

**Cooperative\_Management\_System.java**

import java.util.Scanner;

import javax.swing.JOptionPane;

import javax.swing.JLabel;

import java.util.HashMap;

import java.util.ArrayList;

import java.util.List;

public class Cooperative\_Management\_System {

public static Database db = new Database();

public static void main(String[] args) {

//TEST DATA//

//ACCOUNTS

db.students.put("admin", new Student("admin", "Sherwyne", "Student", "admin"));

db.professors.put("admin", new Professor("admin", "Dr. Sherwyne", "Professor", "admin", 0));

//ITEMS//

//BOOKS

for(int i = 1; i <= 10; i++){

db.books.put("book" + i, new Book("book" + i, "BOOK\_"+i, "Book", 5 + i, 100 + i, "Noli Me", "SHERWYNE", true));

}

//LACE

db.laces.put("lace1", new Lace("lace1", "CCS Lace", "Lace", 50, 100, "CSS", true));

db.laces.put("lace2", new Lace("lace2", "CBFS Lace", "Lace", 50, 100, "CBFS", true));

db.laces.put("lace3", new Lace("lace3", "CTM Lace", "Lace", 50, 100, "CTM", true));

//SCANTRON

db.scantrons.put("scantron1", new Scantron("scantron1", "Scantron", "Scantron", 9999, 5, true));

//

db.uniforms.put("uniform1", new Uniform("uniform1", "CCS Uniform (Male)", "Uniform", 100, 1200, "CCS", true));

db.uniforms.put("uniform2", new Uniform("uniform2", "CCS Uniform (Female)", "Uniform", 100, 1200, "CCS", true));

db.uniforms.put("uniform3", new Uniform("uniform3", "CBFS Uniform (Male)", "Uniform", 100, 2000, "CBFS", true));

db.uniforms.put("uniform4", new Uniform("uniform4", "CBFS Uniform (Female)", "Uniform", 100, 2000, "CBFS", true));

db.uniforms.put("uniform5", new Uniform("uniform5", "CTM Uniform (Male)", "Uniform", 100, 3000, "CTM", true));

db.uniforms.put("uniform6", new Uniform("uniform6", "CTM Uniform (Female)", "Uniform", 100, 3000, "CTM", true));

int main\_choice = 999;

while(main\_choice != -1){

main\_choice = db.main\_menu();

switch(main\_choice){

case 0:

//Login

db.login();

break;

case 1:

//Register

db.register();

break;

}

}

}

}

Item.java

**import java.util.HashMap;**

**import javax.swing.JOptionPane;**

**public class Item {**

**String item\_id, item\_type, item\_name;**

**boolean available;**

**int item\_quantity;**

**double item\_price;**

**public Item(String item\_id, String item\_name, String item\_type, int item\_quantity, double item\_price, boolean available) {**

**this.item\_id = item\_id;**

**this.item\_name = item\_name;**

**this.item\_type = item\_type;**

**this.item\_quantity = item\_quantity;**

**this.item\_price = item\_price;**

**this.available = available;**

**}**

**void update\_id(String new\_value){**

**item\_id = new\_value;**

**}**

**void update\_name(String new\_value){**

**item\_name = new\_value;**

**}**

**void update\_type(String new\_value){**

**item\_type = new\_value;**

**}**

**void update\_quantity(int new\_value){**

**if(new\_value > 0){**

**item\_quantity = new\_value;**

**available = true;**

**}**

**else{**

**item\_quantity = 0;**

**available = false;**

**}**

**}**

**void add\_quantity(int new\_value){**

**item\_quantity += new\_value;**

**available = true;**

**}**

**void update\_price(double new\_value){**

**item\_price = new\_value;**

**}**

**void update\_available(boolean new\_value){**

**available = new\_value;**

**}**

**void buy(double amount\_pay, int buy\_quantity, String user\_type){**

**if(item\_quantity - buy\_quantity >= 0){**

**double total = buy\_quantity \* item\_price;**

**double change = amount\_pay - total;**

**item\_quantity -= buy\_quantity;**

**available = (item\_quantity == 0) ? false : true;**

**if(user\_type.equals("Student")){**

**JOptionPane.showMessageDialog(null, "Transaction Complete! \nChange: " + change);**

**}**

**else{**

**JOptionPane.showMessageDialog(null, "Transaction Complete!");**

**}**

**}**

**else if(item\_quantity - buy\_quantity < 0){**

**JOptionPane.showMessageDialog(null, "Out of Stock!");**

**}**

**}**

**}**

**class Book extends Item{**

**String title, author;**

**public Book(String item\_id, String item\_name, String item\_type, int item\_quantity, double item\_price, String title, String author, boolean available){**

**super(item\_id, item\_name, item\_type, item\_quantity, item\_price, available);**

**this.title = title;**

**this.author = author;**

**}**

**void update\_title(String new\_value){**

**title = new\_value;**

**}**

**void update\_author(String new\_value){**

**author = new\_value;**

**}**

**@Override**

**public String toString(){**

**return item\_id + "," + item\_type + "," + item\_quantity + "," + item\_price + "," + title + "," + author + "," + available;**

**}**

**}**

**class Lace extends Item{**

**String college;**

**public Lace(String item\_id, String item\_name, String item\_type, int item\_quantity, double item\_price, String college, boolean available){**

**super(item\_id, item\_name, item\_type, item\_quantity, item\_price, available);**

**this.college = college;**

**}**

**void update\_college(String new\_value){**

**college = new\_value;**

**}**

**@Override**

**public String toString(){**

**return item\_id + "," + item\_type + "," + item\_quantity + "," + item\_price + "," + college + "," + available;**

**}**

**}**

**class Scantron extends Item{**

**public Scantron(String item\_id, String item\_name, String item\_type, int item\_quantity, double item\_price, boolean available){**

**super(item\_id, item\_name, item\_type, item\_quantity, item\_price, available);**

**}**

**@Override**

**public String toString(){**

**return item\_id + "," + item\_type + "," + item\_quantity + "," + item\_price + "," + available;**

**}**

**}**

**class Uniform extends Item{**

**String college;**

**public Uniform(String item\_id, String item\_name, String item\_type, int item\_quantity, double item\_price, String college, boolean available){**

**super(item\_id, item\_name, item\_type, item\_quantity, item\_price, available);**

**this.college = college;**

**}**

**void update\_college(String new\_value){**

**college = new\_value;**

**}**

**@Override**

**public String toString(){**

**return item\_id + "," + item\_type + "," + item\_quantity + "," + item\_price + "," + college + "," + available;**

**}**

**}**

**User.java**

**import javax.swing.JOptionPane;**

**import java.util.HashMap;**

**import java.util.ArrayList;**

**import java.util.Arrays;**

**import javax.swing.JTable;**

**import javax.swing.JScrollPane;**

**import javax.swing.table.DefaultTableModel;**

**public class User{**

**String user\_id,user\_name,user\_type,password,category;**

**//IF UPDATING THIS ONE MAKE SURE TO UPDATE Database.java**

**//Also update calling of User Object and its Child Class**

**public User(String user\_id, String user\_name, String user\_type, String password){**

**this.user\_id = user\_id;**

**this.user\_name = user\_name;**

**this.user\_type = user\_type;**

**this.password = password;**

**}**

**void update\_name(String value){**

**user\_name = value;**

**}**

**void update\_password(String value){**

**password = value;**

**}**

**}**

**class Professor extends User{**

**static double loan\_balance;**

**public Professor(String user\_id, String user\_name, String user\_type, String password, double loan\_balance){**

**super(user\_id, user\_name, user\_type, password);**

**this.loan\_balance = loan\_balance;**

**}**

**@Override**

**public String toString(){**

**return user\_name + " " + user\_type + " " + loan\_balance;**

**}**

**void check\_balance(){**

**JOptionPane.showMessageDialog(null, "Loan Balance: \u20B1" + loan\_balance);**

**}**

**void add\_loan(){**

**try{**

**String str\_loan = JOptionPane.showInputDialog(null, "Enter Amount to Loan: ");**

**if(str\_loan == null){**

**//DO NOTHING!**

**}**

**else{**

**double amount\_loan = Double.parseDouble(str\_loan);**

**if(amount\_loan == 0){**

**JOptionPane.showMessageDialog(null, "Invalid Input!");**

**add\_loan();**

**}**

**else{**

**double total = loan\_balance + amount\_loan;**

**int confirm = JOptionPane.showConfirmDialog(null, "Do you want to loan \u20B1" + amount\_loan + "? \nTotal Loan: \u20B1" + total,**

**"Confirm Loan",**

**JOptionPane.YES\_NO\_OPTION);**

**if(confirm == JOptionPane.YES\_OPTION){**

**loan\_balance += amount\_loan;**

**JOptionPane.showMessageDialog(null, "Successfully Loaned \u20B1" + total);**

**}**

**total = 0;**

**}**

**}**

**}**

**catch(NumberFormatException e){**

**JOptionPane.showMessageDialog(null, "Invalid Input!");**

**add\_loan();**

**}**

**}**

**void pay\_loan(){**

**if(loan\_balance == 0){**

**JOptionPane.showMessageDialog(null, "No Balance To Pay");**

**}**

**else{**

**try{**

**String str\_pay = JOptionPane.showInputDialog(null, "Total Loan Balance: \u20B1" + loan\_balance +**

**"\nEnter Amount To Pay:");**

**if(str\_pay == null){**

**//DO NOTHING**

**}**

**else{**

**double amount\_pay = Double.parseDouble(str\_pay);**

**if(amount\_pay > 0){**

**int confirm = JOptionPane.showConfirmDialog(null, "Do you want to pay \u20B1" + amount\_pay + "?",**

**"Confirm Payment",**

**JOptionPane.YES\_NO\_OPTION);**

**if(confirm == JOptionPane.YES\_OPTION){**

**double change = amount\_pay - loan\_balance;**

**if(amount\_pay < loan\_balance){**

**change = 0;**

**}**

**loan\_balance -= amount\_pay - change;**

**JOptionPane.showMessageDialog(null, "Successfully Paid \u20B1" + amount\_pay +**

**"\nChange: " + change +**

**"\nTotal Loan Balance: \u20B1" + loan\_balance);**

**}**

**double total = 0;**

**}**

**else{**

**JOptionPane.showMessageDialog(null, "Invalid Input!");**

**pay\_loan();**

**}**

**}**

**}**

**catch(NumberFormatException e){**

**JOptionPane.showMessageDialog(null, "Invalid Input!");**

**pay\_loan();**

**}**

**}**

**}**

**void update\_balance(double value){**

**loan\_balance = value;**

**}**

**}**

**class Student extends User{**

**public Student(String user\_id, String user\_name, String user\_type, String password){**

**super(user\_id, user\_name, user\_type, password);**

**}**

**@Override**

**public String toString(){**

**return user\_name + " " + user\_type;**

**}**

**int view\_items(){**

**String[] item\_type\_menu = {"Book", "Lace", "Scantron", "Uniform"};**

**int choice\_type = JOptionPane.showOptionDialog(null, "Choose Item", "Menu | Item",**

**JOptionPane.DEFAULT\_OPTION,**

**JOptionPane.INFORMATION\_MESSAGE,**

**null,**

**item\_type\_menu, item\_type\_menu[0]);**

**return choice\_type;**

**}**

**void view\_purchase(HashMap<String, ArrayList<ArrayList<Object>>> receipt, String user\_id){**

**Object[] cols = {"Receipt Code", "Item ID", "Item Name", "Quantity", "Price", "Paid", "Change", "Item Type", "Refunded"};**

**DefaultTableModel tableModel;**

**JTable table;**

**tableModel = new DefaultTableModel(cols, 0);**

**try{**

**ArrayList<ArrayList<Object>> list\_purchase\_info = receipt.get(user\_id);**

**for(ArrayList<Object> purchase\_info : list\_purchase\_info){**

**Object receipt\_code = purchase\_info.get(0);**

**Object item\_id = purchase\_info.get(1);**

**Object item\_name = purchase\_info.get(2);**

**Object quantity = purchase\_info.get(3);**

**Object price = purchase\_info.get(4);**

**Object paid = purchase\_info.get(5);**

**Object change = purchase\_info.get(6);**

**Object type = purchase\_info.get(7);**

**Object isRefunded = purchase\_info.get(8);**

**Object[] rows = {receipt\_code, type, item\_id, item\_name, quantity, "\u20B1"+price, "\u20B1"+paid, "\u20B1"+change, isRefunded};**

**tableModel.addRow(rows);**

**}**

**}**

**catch(NullPointerException e){**

**//DO NOTHING**

**}**

**finally{**

**table = new JTable(tableModel);**

**table.setAutoCreateRowSorter(true);**

**JOptionPane.showMessageDialog(null, new JScrollPane(table));**

**}**

**}**

**void generate\_receipt(HashMap<String, ArrayList<ArrayList<Object>>> receipt, String user\_id, String product\_id, String product\_name, int product\_quantity, double product\_price, double paid\_total, double change, String type){**

**int receipt\_count = receipt.size();**

**String gen\_receipt = user\_id + product\_id + "-" + receipt\_count;**

**ArrayList<ArrayList<Object>> list\_purchase\_info = new ArrayList<ArrayList<Object>>();**

**if(receipt.get(user\_id) != null){**

**list\_purchase\_info = receipt.get(user\_id);**

**}**

**ArrayList<Object> purchase\_info = new ArrayList<Object>(){**

**{**

**add(gen\_receipt);**

**add(product\_id);**

**add(product\_name);**

**add(product\_quantity);**

**add(product\_price);**

**add(paid\_total);**

**add(change);**

**add(type);**

**add(false);**

**}**

**};**

**list\_purchase\_info.add(purchase\_info);**

**receipt.put(user\_id, list\_purchase\_info);**

**JOptionPane.showMessageDialog(null, "Successfully Generated Receipt!" + "\nReceipt Code: " + gen\_receipt);**

**}**

**Object[] refund\_item(HashMap<String, ArrayList<ArrayList<Object>>> receipt, String user\_id){**

**try{**

**ArrayList<ArrayList<Object>> receipt\_item= receipt.get(user\_id);**

**if(receipt\_item.size() > 0){**

**String input\_receipt = JOptionPane.showInputDialog(null, "Enter Receipt Code: ");**

**int receipt\_index = -1;**

**for(int i = 0; i < receipt\_item.size(); i++){**

**ArrayList<Object> item = receipt\_item.get(i);**

**Object gen\_receipt = item.get(0);**

**if(gen\_receipt.equals(input\_receipt)){**

**receipt\_index = i;**

**}**

**}**

**if(receipt\_index >= 0){**

**ArrayList<Object> receipt\_details = receipt\_item.get(receipt\_index);**

**Boolean isRefunded = ((Boolean) receipt\_details.get(8)).booleanValue();**

**if(!isRefunded){**

**String receipt\_code = receipt\_details.get(0).toString();**

**String id = receipt\_details.get(1).toString();**

**String item\_name = receipt\_details.get(2).toString();**

**int quantity = ((Integer) receipt\_details.get(3)).intValue();**

**double price = ((Double) receipt\_details.get(4)).doubleValue();**

**double paid = ((Double) receipt\_details.get(5)).doubleValue();**

**double change = ((Double) receipt\_details.get(6)).doubleValue();**

**String type = receipt\_details.get(7).toString();**

**String details = "Receipt Code: " + receipt\_code + "\n" +**

**"Item Type: " + type + "\n" +**

**"Item ID: " + id + "\n" +**

**"Item Name: " + item\_name + "\n" +**

**"Quantity: " + quantity + "\n" +**

**"Item Price: " + "\u20B1"+price + "\n" +**

**"Amount Paid: " + "\u20B1"+paid + "\n" +**

**"Change: " + "\u20B1"+change;**

**int confirm = JOptionPane.showConfirmDialog(null, "Do you want to refund the following? \n" + details, "Confirm Refund", JOptionPane.YES\_NO\_OPTION);**

**if(confirm == JOptionPane.YES\_OPTION){**

**Object item\_id = receipt\_details.get(1);**

**Object item\_type = receipt\_details.get(7);**

**Object item\_quantity = receipt\_details.get(3);**

**receipt\_details.set(8, true);**

**JOptionPane.showMessageDialog(null, "Successfully Refunded!");**

**Object[] output = {item\_id, item\_type, item\_quantity};**

**return output;**

**}**

**}**

**else{**

**JOptionPane.showMessageDialog(null, "Item Already Refunded!");**

**}**

**}**

**else{**

**JOptionPane.showMessageDialog(null, "Receipt Not Found!");**

**}**

**}**

**}**

**catch(NullPointerException e){**

**//DO NOTHING**

**JOptionPane.showMessageDialog(null, "No Items To Refund!");**

**}**

**return null;**

**}**

**}**

**class Admin extends User{**

**public Admin(String user\_id, String user\_name, String user\_type, String password){**

**super(user\_id, user\_name, user\_type, password);**

**}**

**//USER MANAGEMENT**

**void manage\_user(){**

**JOptionPane.showMessageDialog(null, "MANAGE USER");**

**}**

**//CONTROLS USER**

**void view\_user(){**

**JOptionPane.showMessageDialog(null, "VIEW USER");**

**}**

**void update\_user(){**

**JOptionPane.showMessageDialog(null, "UPDATE USER");**

**}**

**void delete\_user(){**

**JOptionPane.showMessageDialog(null, "DELETE USER");**

**}**

**//CONTROLS ITEM**

**int menu\_item(){**

**String[] item\_type\_menu = {"Book", "Lace", "Scantron", "Uniform"};**

**int choice\_type = JOptionPane.showOptionDialog(null, "Choose Item", "Menu | Item",**

**JOptionPane.DEFAULT\_OPTION,**

**JOptionPane.INFORMATION\_MESSAGE,**

**null,**

**item\_type\_menu, item\_type\_menu[0]);**

**return choice\_type;**

**}**

**//CONTROLS USER**

**int menu\_user(){**

**String[] user\_type\_menu = {"Student", "Professor"};**

**double loan\_balance;**

**int choice\_type = JOptionPane.showOptionDialog(null, "Choose User", "Menu | User",**

**JOptionPane.DEFAULT\_OPTION,**

**JOptionPane.INFORMATION\_MESSAGE,**

**null,**

**user\_type\_menu, user\_type\_menu[0]);**

**return choice\_type;**

**}**

**}**