**Advanced Programming**

(CSCI207)

**Pharmacy ordering project**  
  
Team Name

19-2000

Members

Amr Shaarawy | 211001149

Omar Adel | 211000540

Yassin Elasfar | 211002150

Mamdouh koritm | 211001149

Table of Contents

[**Abstract** 3](#_Toc135509327)

[**Introduction** 3](#_Toc135509328)

[**Methodology** 4](#_Toc135509329)

[**Snapshots** 7](#_Toc135509330)

[**UML** 8](#_Toc135509331)

[**Result** 9](#_Toc135509332)

# **Abstract**

The Pharmacy Ordering Project is a menu-driven program designed to enable users to order drugs from a pharmacy. The program categorizes drugs into three categories: cosmetics, prescription drugs, and other. Users can add drugs to the pharmacy's inventory, remove drugs, place orders, and retrieve the total sales for the day. The program ensures that the pharmacy does not exceed its capacity while adding new drugs. The ordering process involves searching for the drug by its ID, getting its unit price, printing the price, and asking for a prescription if necessary. At the end of each day, the program calculates the total sales made.

# **Introduction**

The Pharmacy Ordering Project is designed to simplify the process of ordering drugs from a pharmacy. The program is menu-driven, enabling users to add new drugs to the pharmacy's inventory, remove drugs, place orders, and retrieve the total sales made. The program categorizes drugs into three categories: cosmetics, prescription drugs, and other.

Upon program execution, the user is prompted to enter the pharmacy's capacity, i.e., the number of drugs the pharmacy can hold. The program ensures that the pharmacy does not exceed its capacity while adding new drugs. The user can add drugs to the pharmacy's inventory by providing the drug's name, ID, price, category, and available quantity.

To place an order, the user enters the drug's ID, and the program searches for the drug's availability. If the drug is available, the program retrieves its unit price and prints it for the user. If the drug is a prescription drug, the user is prompted to provide the prescription. For cosmetics, the program applies a markup of 20% (1.2 times the original price).

At the end of each day, the program calculates the total sales made by storing the quantity and total price of each order. The user can retrieve the total sales for the day by selecting the appropriate option from the main menu.

The Pharmacy Ordering Project simplifies the ordering process for both the pharmacy and the customers. It ensures that the pharmacy's inventory stays within its capacity and provides real-time information on the availability and pricing of drugs. The program's user-friendly interface makes it easy for customers to place orders and retrieve information, making the ordering process quicker and more efficient.

# **Methodology**

The Pharmacy Ordering Project consists of two main classes: the Drug class and the Pharmacy class. The Drug class represents a drug and its properties, while the Pharmacy class represents the pharmacy and its inventory.

The Drug class has five private instance variables: name, id, price, category, and quantity. The constructor initializes these variables, and there are getter and setter methods for each variable. The category variable is an integer that represents the drug's category, where 1 represents prescription drugs and 2 represents cosmetics. The getCategory() method returns a Boolean indicating whether the drug is a cosmetic or not.

The Pharmacy class has an ArrayList of Drug objects, representing the pharmacy's inventory. It also has three private instance variables: capacity, totalSales, and quantitySold. The constructor initializes the capacity, totalSales, and quantitySold variables. There are methods to add and remove drugs from the inventory, check for duplicate IDs, and place orders. The addDrug() method adds a new drug to the inventory if there is enough capacity, and the removeDrug() method removes a drug from the inventory by its ID. The placeOrder() method searches for the drug by its ID, checks if it is available and the user has the appropriate prescription, calculates the total price, and updates the total sales and quantity sold. The getTotalSales() method returns the total sales and quantity sold for the day.

The program utilizes a menu-driven approach, allowing users to choose from various options to add drugs to the inventory, remove drugs, place orders, retrieve the total sales for the day, and exit the program. The program prompts the user to enter the pharmacy's capacity when it first runs and ensures that the pharmacy does not exceed its capacity while adding new drugs.

The Drug and Pharmacy classes provide a modular and object-oriented approach to the Pharmacy Ordering Project, making it more organized and easier to maintain. The classes encapsulate data and behavior, allowing for better control and security of the program's functionality.

The Pharmacy Ordering Project is split into different frames, each of which has a specific job. The frames are as follows:

Frame 1:

The first frame is the welcome screen that asks the user to enter the pharmacy's capacity. It has a text field and an OK button. When the user enters the capacity and clicks the OK button, the program creates a new instance of the Pharmacy class with the specified capacity and opens the main menu in the next frame.

Frame 2:

The second frame is the main menu and provides several options to the user. It has five buttons: Add Drug, Remove Drug, Place an Order, Get Total Sales, and Exit. When the user clicks on any of these buttons, a new frame opens up corresponding to the selected option.

Add Drug Frame:

The Add Drug frame allows the user to add a new drug to the pharmacy's inventory. It has several text fields for the drug's name, ID, price, category, and quantity, as well as a Submit button. When the user enters this information and clicks the Submit button, the program adds the new drug to the inventory if there is enough capacity and returns to the main menu frame.

Remove Drug Frame:

The Remove Drug frame allows the user to remove a drug from the pharmacy's inventory. It has a text field for the drug's ID and a Submit button. When the user enters the ID and clicks the Submit button, the program removes the drug from the inventory if it exists and returns to the main menu frame.

Place an Order Frame:

The Place an Order frame allows the user to place an order for a drug. It has a text field for the drug's ID and a Submit button. When the user enters the ID and clicks the Submit button, the program checks if the drug is available and the user has the appropriate prescription. If yes, the program calculates the total price and updates the total sales and quantity sold. Then, the program returns to the main menu frame.

Get Total Sales Frame:

The Get Total Sales frame allows the user to retrieve the total sales for the day. It has a button to press to show the sales and a return button to go back to the main menu. When the user clicks the Show Sales button, the program displays the total sales and quantity sold for the day. When the user clicks the Return button, the program returns to the main menu frame.

By dividing the Pharmacy Ordering Project into different frames, each with a specific job, the program becomes more organized and easier to use. The frames provide a user-friendly interface that simplifies the ordering process and makes it quicker and more efficient. The Pharmacy Ordering Project is a great example of how a graphical user interface (GUI) can enhance the functionality of a program and make it more accessible to users.

# **Snapshots**

A screenshot of a computer

Description automatically generatedA screenshot of a computer menu

Description automatically generated with medium confidence

A screenshot of a computer error

Description automatically generated with medium confidenceA screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated with medium confidenceA screenshot of a computer

Description automatically generated with medium confidence

# **UML**

|  |
| --- |
| **Drug** |
| - name: String  - id: int  - price: double  - category: int  - quantity: int  - prescriptionRequired: boolean |
| + Drug(name: String, id: int, price: double, category: String, quantity: int, prescription: boolean)  + getName(): String  + getId(): int  + getPrice(): double  + getCategory(): String  + getQuantity(): int  + isPrescriptionRequired(): boolean  + setQuantity(quantity: int): void |

|  |
| --- |
| **PharmacySystem** |
| - capacity: int  - drugs: List<Drug>  - totalSales: double  - frame: JFrame - menuPanel: JPanel |
| + PharmacySystem()  + initializeUI(): void  + displayMenu(): void  + addDrug(name: String, id: int, price: double, category: String, quantity: int): void  + removeDrug(id: int): void  + placeOrder(id: int): void |  +calculateTotalSales(): double |

|  |
| --- |
| **JFrame** |
| + JFrame()  + setPanel(panel:JPanel): void |

|  |
| --- |
| **JPanel** |
| + addComponent(component: Component): void  +removeComponent(component: Component): void |

# **Result**

The Pharmacy Ordering Project is a menu-driven program designed to simplify the ordering process for drugs from a pharmacy. The program categorizes drugs into three categories: cosmetics, prescription drugs, and other. Users can add drugs to the pharmacy's inventory, remove drugs, place orders, and retrieve the total sales for the day.

The program utilizes a modular and object-oriented approach, with the Drug and Pharmacy classes providing encapsulation of data and behavior, ensuring better control and security of the program's functionality. The program's user-friendly interface, divided into different frames, makes it easy for customers to place orders and retrieve information, enhancing the ordering process's speed and efficiency.

The Pharmacy Ordering Project is a great example of how a graphical user interface (GUI) can enhance the functionality of a program and make it more accessible to users. It simplifies the ordering process for both the pharmacy and its customers, providing real-time information on drug availability and pricing while ensuring that the pharmacy's inventory stays within its capacity. The program's ability to calculate the total sales for the day makes it easy for the pharmacy to keep track of its sales and inventory.

In conclusion, the Pharmacy Ordering Project is a well-designed program that simplifies the ordering process for drugs from a pharmacy. Its modular and object-oriented approach, along with its user-friendly interface, makes it a great example of how GUI can enhance the functionality of a program and make it more accessible to users.