A blue text on a black background

Description automatically generatedA colorful triangles on a black background

Description automatically generated

Software Development Assignment 2

Robert Leniston

B00158877

Department of Informatics

School of Informatics and Engineering

TU Dublin Blanchardstown Campus

Dublin 15.

Higher Certificate in Science in

Computer Systems

Software Development 2

14/04/2024

# Table of contents

Table of Contents

[Table of contents 2](#_Toc164098983)

[Program Design 3](#_Toc164098984)

[Program challenges 4](#_Toc164098985)

[*Try/Catch* 4](#_Toc164098986)

[Item selection confirmation 4](#_Toc164098987)

[Confirmation of item added to cart 5](#_Toc164098988)

[Changing font colour 5](#_Toc164098989)

[UI Screenshots 7](#_Toc164098990)

# Program Design

The purpose of this program is to simulate a Point of Sale (POS) system for a hardware store.

The application allows the user to view available item, add the items to their cart, view their cart and finally checkout. The application uses file I/O to load the stock items from a text file and then to their transactions in another text file.

The application is split up into several sections, which each section performing its own unique function.

1. public class PointOfSaleSystem {
   * Purpose: Marks the beginning of the main class where programs functionality is initiated
2. public static void main(String[] args):
   * Purpose: This is the entry point of the program. It initializes the application, loads stock items, displays the menu, and handles user input.
3. public static void displayMenu():
   * Purpose: Displays the menu options for the user to interact with.
4. private static void loadStockItems():
   * Purpose: Loads the stock items from the stock.txt file into the program's memory.
5. public static void viewItems():
   * Purpose: Displays the available items in the stock list.
6. public static void addItemToCart(Scanner scanner):
   * Purpose: Adds selected items to the shopping cart.
7. public static void viewCart():
   * Purpose: Displays the items currently in the shopping cart.
8. public static void checkout():
   * Purpose: Processes the checkout, calculates the total amount, writes transaction details to a file, and clears the shopping cart.

# Program challenges

## Try/Catch

When implementing Try/Catch methods in my code to handle the potential I/O exception errors, I learned of the Try-with-resource method. I decided to implement the method over regular Try/Catch for several reasons.

The Try-Catch method required manual closing of resources which can easily be forgotten, whereas Try-with-resource will automatically close resources after the block of code is finished. This helps to eliminate resource leaks which can be very problematic. Try-with-resource also allowed for easier to read and cleaner code.

## Item selection confirmation

Originally, when an item was chosen to be added to the basket, the user was sent back to the main menu. I wanted to change this so that once an option is selected, the user is then prompted once more to enter another item to add to their cart.

This was achieved by the introduction of the ‘return’ statement in the ‘addItemToCart’ method. Originally, I included a break statement to exit the loop when a user chose an item to add to their basket.

// Method to add an item to the cart  
private static void addItemToCart(Scanner scanner) {  
 int itemNumber;  
 do {  
 System.*out*.print("Enter the item number to add to cart (or 0 to exit): ");  
 itemNumber = scanner.nextInt();  
 if (itemNumber == 0) {  
 break; // Exit the loop if the user enters 0  
 } else if (itemNumber >= 1 && itemNumber <= *stockItems*.size()) {  
 String itemToAdd = *stockItems*.get(itemNumber - 1);  
 *cart*.add(itemToAdd);  
 System.*out*.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");  
 System.*out*.println("Item added to cart: " + itemToAdd);  
 System.*out*.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");  
 System.*out*.println();  
 // After adding the item, break out of the loop to return to the main menu  
 break;  
 } else {  
 System.*out*.println("Invalid item number.");  
 }  
 } while (true);  
}

This code sent the user back to the main menu after choosing an option. The code segment with the return statement enabled the user to stay within the ‘addItemToCart’ menu.

private static void addItemToCart(Scanner scanner) {  
 int itemNumber;  
 do {  
 System.*out*.print("Enter the item number to add to cart (or 0 to exit): ");  
 itemNumber = scanner.nextInt();  
 if (itemNumber == 0) {  
 return; // Exit the method if the user enters 0  
 } else if (itemNumber >= 1 && itemNumber <= *stockItems*.size()) {  
 String itemToAdd = *stockItems*.get(itemNumber - 1);  
 *cart*.add(itemToAdd);  
 System.*out*.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");  
 System.*out*.println("Item added to cart: " + itemToAdd);  
 System.*out*.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");  
 System.*out*.println();  
 } else {  
 System.*out*.println("Invalid item number.");  
 }  
 } while (true);  
}

## Confirmation of item added to cart

A problem I encountered in my application was a lack of feedback when a user selected an item from the stock list to add to the basket. When I user chose an option they received no feedback, they were just prompted to choose another item from the list.

System.*out*.println("Item added to cart: " + itemToAdd);

I wanted a message to appear informing the user of what it was that they just added to their basket. To achieve this, I implemented a simple but effective System.out.print concatenated with variable. This allows the user to know what it is they have added to their basked, it was a small change, but it led to a much more user friendly interface.

## Changing font colour

To improve the readability of my assignment, I sought to change the colour of the text output that the user would read. This would ensure that any information that the user should receive is clearly marked and that nothing should slip through the tracks.

To achieve this, I researched methods on how coloured text output in java is done. My research pointed me towards ANSI escape code. The constants “RED” & “RESET” were defined within the PointOfSale Class, they are string constants containing ANSI escape sequences for changing the text colour.

RED and RESET are then concatenated with the desired text output to achieve the coloured text.

public static final String RED = "\u001B[31m";  
public static final String RESET = "\u001B[0m";

System.out.println(**RED +**"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");  
System.out.println("Transaction completed. Thank you!");  
System.out.println("Please find your receipt in the Transaction file");  
System.out.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" **+ RESET**);  
System.out.println("");

A screenshot of a computer

Description automatically generated

# A screenshot of a computer Description automatically generatedUI Screenshots

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated