



Programming Fundamentals 1

Produced by Mr. Dave Drohan (david.drohan@setu.ie)
Dr. Siobhán Drohan

Ms. Mairead Meagher

Department of Computing & Mathematics
South East Technological University
Waterford, Ireland

setu.ie





SHOP V2.2

An Array of Products + a Menu

Basic Menu

```
Shop Menu
-----
1) List the Products
2) List the current products
3) Display average product unit cost
4) Display cheapest product
5) List products that are more expensi
0) Exit
==>> 4
The cheapest product is: Product 2

Press any key to continue...
|
```

ShopV2.2 · Menu Driven
Console App



Agenda

☐ RECAP : SHOP V2.0

☐ SHOP V2.1

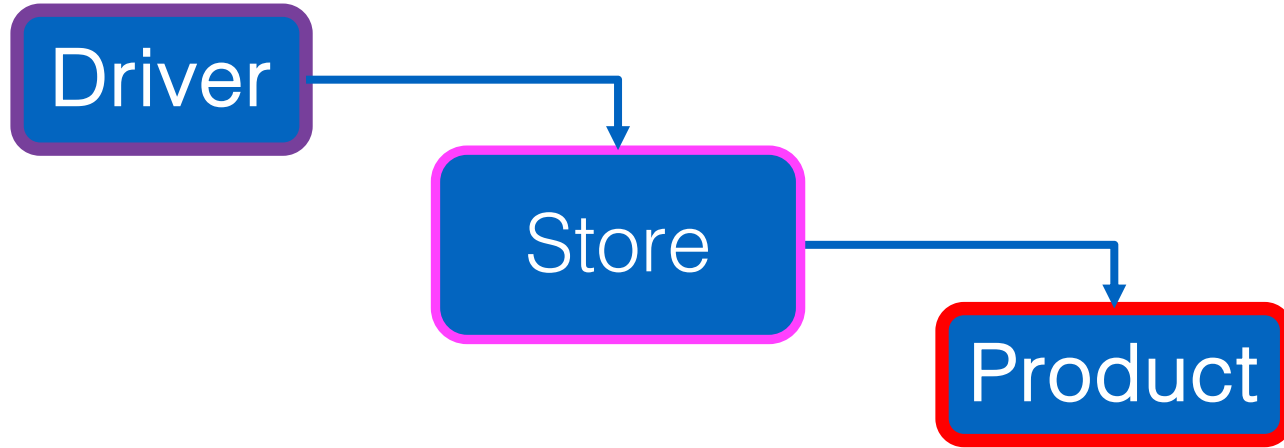
☐ SHOP V2.2



RECAP : SHOP V2.0



Recap - Shop V2.0 - Product

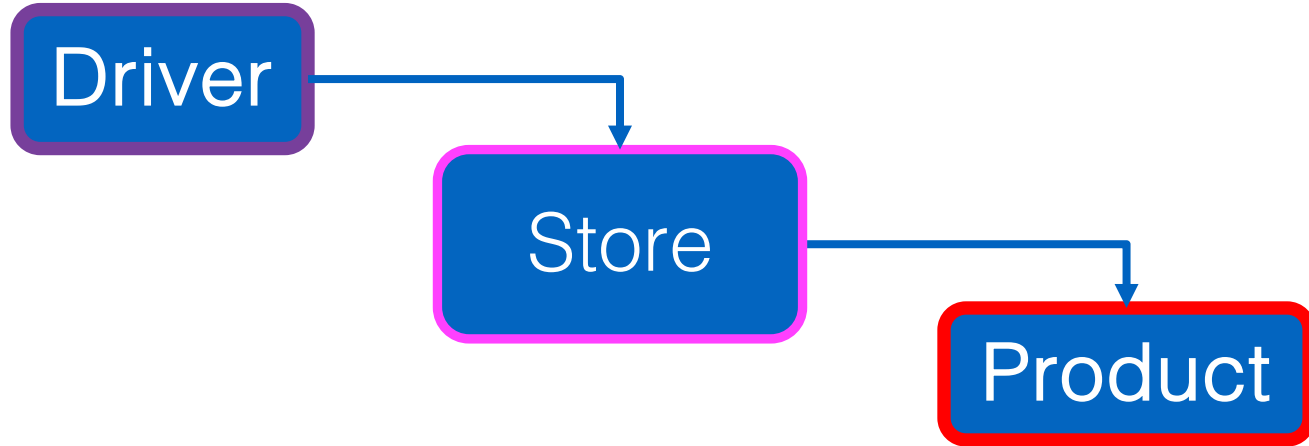


- The **Product** class stores **details** about a product
 - name, code, unit cost, in the current product line



Recap - Shop V2.0

- ❑ New **Store** class is responsible for maintaining a collection of Products
 - i.e. an **array** of Products.
- ❑ **Driver** will now allow the user to decide **how many product** details they want to store.





SHOP V2.1

Version
Developed in
Lab Exercises



Shop V2.1 – Lab Exercises



```
How many Products would you like to have in your Store? 3
Enter the Product Name: Product1
Enter the Product Code: 1
Enter the Unit Cost: 45.99
Is this product in your current line (y/n): Y

Enter the Product Name: Product2
Enter the Product Code: 2
Enter the Unit Cost: 12.99
Is this product in your current line (y/n): N

Enter the Product Name: Product3
Enter the Product Code: 3
Enter the Unit Cost: 23.50
Is this product in your current line (y/n): Y

List of Products are:
0: Product description: Product1, product code: 1, unit cost: 45.99, currently in product line: true
1: Product description: Product2, product code: 2, unit cost: 12.99, currently in product line: false
2: Product description: Product3, product code: 3, unit cost: 23.5, currently in product line: true

List of CURRENT Products are:
0: Product description: Product1, product code: 1, unit cost: 45.99, currently in product line: true
2: Product description: Product3, product code: 3, unit cost: 23.5, currently in product line: true

The average product price is: 27.493333333333336
The cheapest product is: Product2
View the product costing more than this price: 12.99
0: Product description: Product1, product code: 1, unit cost: 45.99, currently in product line: true
2: Product description: Product3, product code: 3, unit cost: 23.5, currently in product line: true
```


Shop V2.1 – Lab Exercises



```
How many Products would you like to have in your Store? 3
Enter the Product Name: Product1
Enter the Product Code: 1
Enter the Unit Cost: 45.99
Is this product in your current line (y/n): Y

Enter the Product Name: Product2
Enter the Product Code: 2
Enter the Unit Cost: 12.99
Is this product in your current line (y/n): N

Enter the Product Name: Product3
Enter the Product Code: 3
Enter the Unit Cost: 23.50
Is this product in your current line (y/n): Y

List of Products are:
0: Product description: Product1, product code: 1, unit cost: 45.99, currently in product line: true
1: Product description: Product2, product code: 2, unit cost: 12.99, currently in product line: false
2: Product description: Product3, product code: 3, unit cost: 23.5, currently in product line: true

List of CURRENT Products are:
0: Product description: Product1, product code: 1, unit cost: 45.99, currently in product line: true
2: Product description: Product3, product code: 3, unit cost: 23.5, currently in product line: true

The average product price is: 27.493333333333336
The cheapest product is: Product2
View the product costing more than this price: 12.99
0: Product description: Product1, product code: 1, unit cost: 45.99, currently in product line: true
2: Product description: Product3, product code: 3, unit cost: 23.5, currently in product line: true
```

```
public class Driver{
    //code omitted
    public static void main(String[] args) {
        Driver driver = new Driver();
        driver.processOrder();
        driver.printProducts();
        driver.printCurrentProducts();
        driver.printAverageProductPrice();
        driver.printCheapestProduct();
        driver.printProductsAboveAPrice();
    }
    //code omitted
}
```



Shop V2.1 – Lab Exercises

Our users have no control of the system; they cannot **choose** to do anything!

```
How many Products would you like to have in your Store? 3
Enter the Product Name: Product1
Enter the Product Code: 1
Enter the Unit Cost: 45.99
Is this product in your current line (y/n): Y

Enter the Product Name: Product2
Enter the Product Code: 2
Enter the Unit Cost: 12.99
Is this product in your current line (y/n): N

Enter the Product Name: Product3
Enter the Product Code: 3
Enter the Unit Cost: 23.50
Is this product in your current line (y/n): Y

List of Products are:
0: Product description: Product1, product code: 1, unit cost: 45.99, currently in product line: true
1: Product description: Product2, product code: 2, unit cost: 12.99, currently in product line: false
2: Product description: Product3, product code: 3, unit cost: 23.5, currently in product line: true

List of CURRENT Products are:
0: Product description: Product1, product code: 1, unit cost: 45.99, currently in product line: true
2: Product description: Product3, product code: 3, unit cost: 23.5, currently in product line: true

The average product price is: 27.493333333333336
The cheapest product is: Product2
View the product costing more than this price: 12.99
0: Product description: Product1, product code: 1, unit cost: 45.99, currently in product line: true
2: Product description: Product3, product code: 3, unit cost: 23.5, currently in product line: true
```

```
public class Driver{

    //code omitted

    public static void main(String[] args) {
        Driver driver = new Driver();
        driver.processOrder();
        driver.printProducts();
        driver.printCurrentProducts();
        driver.printAverageProductPrice();
        driver.printCheapestProduct();
        driver.printProductsAboveAPrice();
    }

    //code omitted
}
```



SHOP V2.2

Adding a
Menu
System





Shop V2.2 – Control with menu

```
How many Products would you like to have in your Store? 3
Enter the Product Name: Product 1
Enter the Product Code: 1234
Enter the Unit Cost: 12.99
Is this product in your current line (y/n): y
Enter the Product Name: Product 2
Enter the Product Code: 2345
Enter the Unit Cost: 7.99
Is this product in your current line (y/n): n
Enter the Product Name: Product 3
Enter the Product Code: 6745
Enter the Unit Cost: 49.99
Is this product in your current line (y/n): y
```

```
Shop Menu
```

```
-----
```

- 1) List the Products
- 2) List the current products
- 3) Display average product unit cost
- 4) Display cheapest product
- 5) List products that are more expensive than a given price
- 0) Exit

```
==>>
```

We are going to add a simple menu that will allow us to view details about the entered products.



Shop V2.2 – Control with menu

```
Shop Menu
-----
 1) List the Products
 2) List the current products
 3) Display average product unit cost
 4) Display cheapest product
 5) List products that are more expensive than a given price
 0) Exit
==>> 1
List of Products are:
0: Product description: Product 1, product code: 1234, unit cost: 12.99, currently in product line: true
1: Product description: Product 2, product code: 2345, unit cost: 7.99, currently in product line: false
2: Product| description: Product 3, product code: 6745, unit cost: 49.99, currently in product line: true

Press any key to continue...
```

Option 1:
List the products





Shop V2.2 – Control with menu

```
Shop Menu
-----
1) List the Products
2) List the current products
3) Display average product unit cost
4) Display cheapest product
5) List products that are more expensive than a given price
0) Exit
==>> 2
List of CURRENT Products are:
0: Product description: Product 1, product code: 1234, unit cost: 12.99, currently in product line: true
2: Product description: Product 3, product code: 6745, unit cost: 49.99, currently in product line: true

Press any key to continue...
```

Option 2:
List the **current** products





Shop V2.2 – Control with menu

```
Shop Menu
-----
 1) List the Products
 2) List the current products
 3) Display average product unit cost
 4) Display cheapest product
 5) List products that are more expensive than a given price
 0) Exit
==>> 3
The average product price is: 23.6566666666666666

Press any key to continue...
|
```



Option 3:
Display average cost



Shop V2.2 – Control with menu

```
Shop Menu
```

```
-----
```

- 1) List the Products
- 2) List the current products
- 3) Display average product unit cost
- 4) Display cheapest product
- 5) List products that are more expensive than a given price
- 0) Exit

```
==>> 4
```

```
The cheapest product is: Product 2
```

```
Press any key to continue...
```



Option 4:

Display cheapest
product



Shop V2.2 – Control with menu

```
Shop Menu
```

```
-----
```

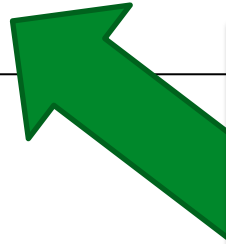
- 1) List the Products
- 2) List the current products
- 3) Display average product unit cost
- 4) Display cheapest product
- 5) List products that are more expensive than a given price
- 0) Exit

```
==>> 5
```

```
View the product costing more than this price: 15
```

```
2: Product description: Product 3, product code: 6745, unit cost: 49.99, currently in product line: true
```

```
Press any key to continue...
```



Option 5:

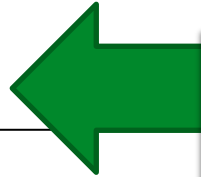
List products that are more expensive than a given price



Shop V2.2 – Control with menu

```
Shop Menu
-----
 1) List the Products
 2) List the current products
 3) Display average product unit cost
 4) Display cheapest product
 5) List products that are more expensive than a given price
 0) Exit
==>> 6
Invalid option entered: 6

Press any key to continue...
```



Invalid Option

Menu is redisplayed once you press the enter key.



Shop V2.2 – Control with menu

```
Press any key to continue...
```

```
Shop Menu
```

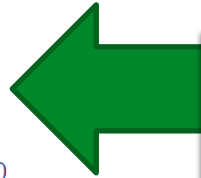
```
-----
```

- 1) List the Products
- 2) List the current products
- 3) Display average product unit cost
- 4) Display cheapest product
- 5) List products that are more expensive than a given price
- 0) Exit

```
==>> 0
```

```
Exiting... bye
```

```
Process finished with exit code 0
```



Option 0:

Exit the system



The Switch Statement



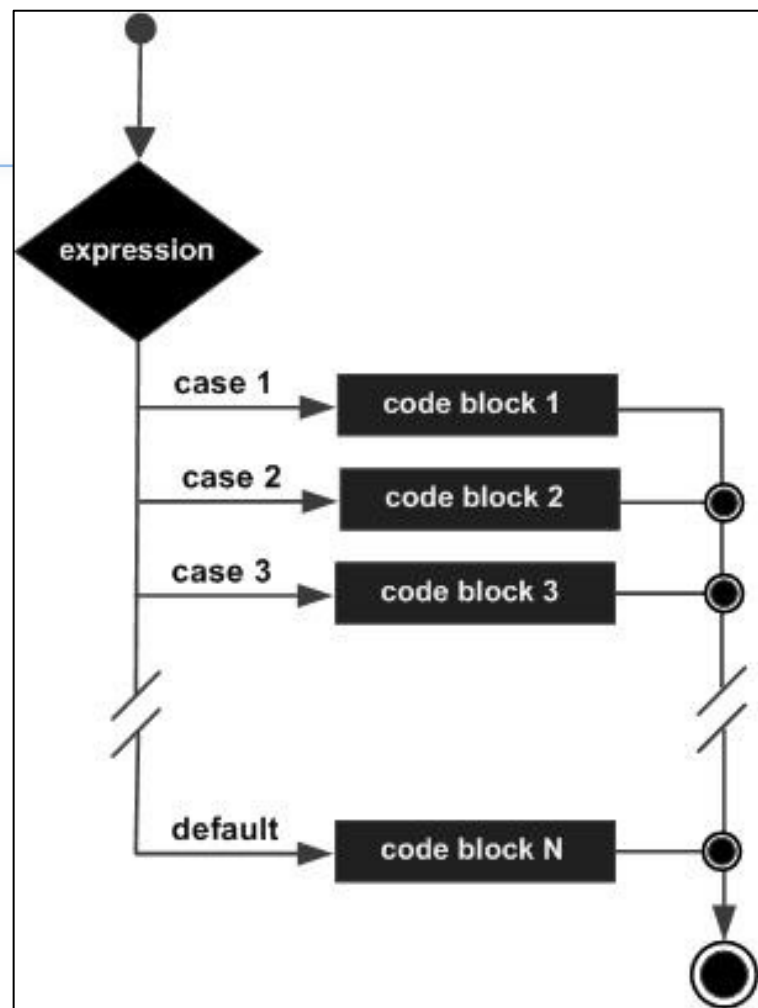
A brief
introduction
(more later..)



The **switch** statement

- The **switch** statement works in exactly the same way as a **set of if** statements, but is more compact and readable
- The **switch statement** switches on a single **value** to one of an arbitrary number of **cases**

The switch statement





The switch statement

- ❑ A *switch* statement can have any number of **case** labels.
- ❑ The **default** case is optional; if no default is given, it may happen that no case is executed.
- ❑ Can *switch* on **int**, **char** or **String**.
- ❑ Let's use this now in ShopV2.2



SHOP V2.2

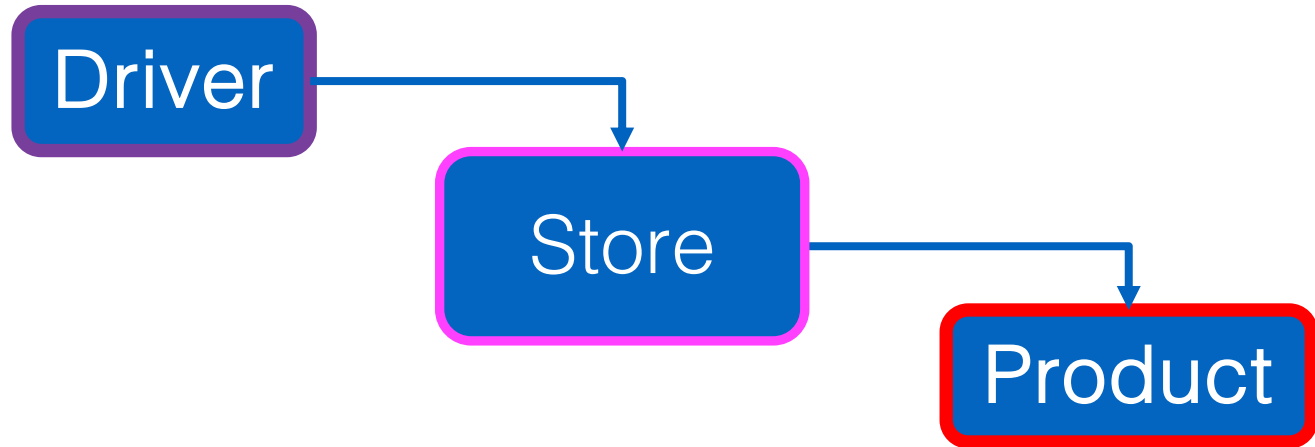
Adding the
Menu
System





SHOP V2.2

- ❑ **Product** – no changes
- ❑ **Store** – no changes
- ❑ **Driver** will be changed to allow the user to choose options from a menu.





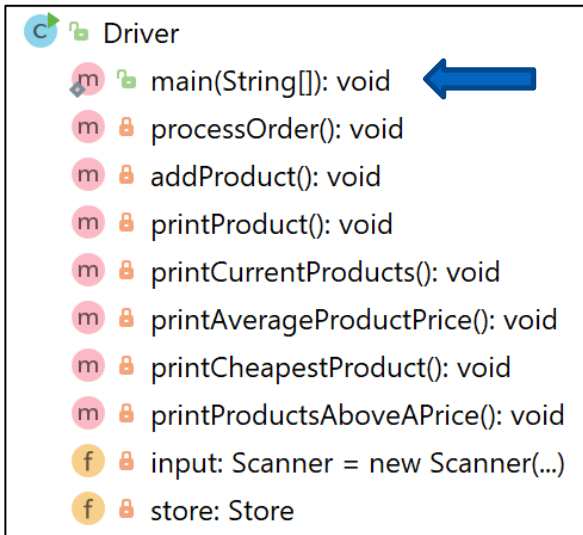
```
Driver
  m main(String[]): void
  m processOrder(): void
  m addProduct(): void
  m printProduct(): void
  m printCurrentProducts(): void
  m printAverageProductPrice(): void
  m printCheapestProduct(): void
  m printProductsAboveAPrice(): void
  f input: Scanner = new Scanner(...)
  f store: Store
```





















```
Driver
  m main(String[]): void
  m processOrder(): void
  m addProduct(): void
  m printProduct(): void
  m printCurrentProducts(): void
  m printAverageProductPrice(): void
  m printCheapestProduct(): void
  m printProductsAboveAPrice(): void
  f input: Scanner = new Scanner(...)
  f store: Store
```

```
Driver
  m Driver()
  m main(String[]): void
  m mainMenu(): int
  m runMenu(): void
  m processOrder(): void
  m addProduct(): void
  m printProduct(): void
  m printCurrentProducts(): void
  m printAverageProductPrice(): void
  m printCheapestProduct(): void
  m printProductsAboveAPrice(): void
  f input: Scanner = new Scanner(...)
  f store: Store
```



Shop V2.1 – main method

 Driver

-   main(String[]): void ←
-   processOrder(): void
-   addProduct(): void
-   printProduct(): void
-   printCurrentProducts(): void
-   printAverageProductPrice(): void
-   printCheapestProduct(): void
-   printProductsAboveAPrice(): void
-   input: Scanner = new Scanner(...)
-   store: Store

```
public static void main(String[] args) {  
    Driver driver = new Driver();  
    driver.processOrder();  
    driver.printProducts();  
    driver.printCurrentProducts();  
    driver.printAverageProductPrice();  
    driver.printCheapestProduct();  
    driver.printProductsAboveAPrice();  
}
```



Shop V2.1 – main method

```
Driver
m main(String[]): void
m processOrder(): void
m addProduct(): void
m printProduct(): void
m printCurrentProducts(): void
m printAverageProductPrice(): void
m printCheapestProduct(): void
m printProductsAboveAPrice(): void
f input: Scanner = new Scanner(...)
f store: Store
```

```
public static void main(String[] args) {
    Driver driver = new Driver();
    driver.processOrder();
    driver.printProducts();
    driver.printCurrentProducts();
    driver.printAverageProductPrice();
    driver.printCheapestProduct();
    driver.printProductsAboveAPrice();
}
```

Console Output

```
How many Products would you like to have in your Store? 3
Enter the Product Name: Product1
Enter the Product Code: 1
Enter the Unit Cost: 45.99
Is this product in your current line (y/n): Y

Enter the Product Name: Product2
Enter the Product Code: 2
Enter the Unit Cost: 12.99
Is this product in your current line (y/n): N

Enter the Product Name: Product3
Enter the Product Code: 3
Enter the Unit Cost: 23.50
Is this product in your current line (y/n): Y

List of Products are:
0: Product description: Product1, product code: 1, unit cost: 45.99, currently in product line: true
1: Product description: Product2, product code: 2, unit cost: 12.99, currently in product line: false
2: Product description: Product3, product code: 3, unit cost: 23.5, currently in product line: true

List of CURRENT Products are:
0: Product description: Product1, product code: 1, unit cost: 45.99, currently in product line: true
2: Product description: Product3, product code: 3, unit cost: 23.5, currently in product line: true

The average product price is: 27.493333333333336
The cheapest product is: Product2
View the product costing more than this price: 12.99
0: Product description: Product1, product code: 1, unit cost: 45.99, currently in product line: true
2: Product description: Product3, product code: 3, unit cost: 23.5, currently in product line: true
```



Shop V2.1 – main method

```
Driver
  m main(String[]): void
  m processOrder(): void
  m addProduct(): void
  m printProduct(): void
  m printCurrentProducts(): void
  m printAverageProductPrice(): void
  m printCheapestProduct(): void
  m printProductsAboveAPrice(): void
  f input: Scanner = new Scanner(...)
  f store: Store
```

```
public static void main(String[] args) {
    Driver driver = new Driver();
    driver.processOrder();
    driver.printProducts();
    driver.printCurrentProducts();
    driver.printAverageProductPrice();
    driver.printCheapestProduct();
    driver.printProductsAboveAPrice();
}
```

These methods will be in a menu system in V2.2



```
public Driver() {  
    processOrder();  
    runMenu();  
}  
  
public static void main(String[] args) {  
    new Driver();  
}
```

Shop V2.2:

- new Driver constructor
- changes to the main method

1

Driver

- m Driver() ← green arrow
- m main(String[]): void ← purple arrow
- m mainMenu(): int ← green arrow
- m runMenu(): void ← green arrow
- m processOrder(): void
- m addProduct(): void
- m printProduct(): void
- m printCurrentProducts(): void
- m printAverageProductPrice(): void
- m printCheapestProduct(): void
- m printProductsAboveAPrice(): void
- f input: Scanner = new Scanner(...)
- f store: Store



```
private int mainMenu(){
    System.out.print("""
    Shop Menu
    -----
    1) List the Products
    2) List the current products
    3) Display average product unit cost
    4) Display cheapest product
    5) List products that are more expensive than a given price
    0) Exit
    ==>> """);
    int option = input.nextInt();
    return option;
}
```

Driver

- m Driver() ←
- m main(String[]): void ←
- m mainMenu(): int ←
- m runMenu(): void ←
- m processOrder(): void
- m addProduct(): void
- m printProduct(): void
- m printCurrentProducts(): void
- m printAverageProductPrice(): void
- m printCheapestProduct(): void
- m printProductsAboveAPrice(): void
- f input: Scanner = new Scanner(...)
- f store: Store

Shop V2.2 – new
mainMenu
method

2



V2.2

```
private void runMenu(){
    int option = mainMenu();

    while (option != 0){
        switch (option){
            case 1 -> printProducts();
            case 2 -> printCurrentProducts();
            case 3 -> printAverageProductPrice();
            case 4 -> printCheapestProduct();
            case 5 -> printProductsAboveAPrice();
            default -> System.out.println("Invalid option entered: " + option);
        }

        //pause the program so that the user can read what we just printed to the terminal window
        System.out.println("\nPress enter key to continue...");
        input.nextLine();
        input.nextLine(); //second read is required - bug in Scanner class; a String read is ignored straight after reading an int.

        //display the main menu again
        option = mainMenu();
    }

    //the user chose option 0, so exit the program
    System.out.println("Exiting...bye");
    System.exit(0);
}
```

3

Shop V2.2 – new
runMenu method



V2.2

```
private void runMenu(){
    int option = mainMenu();

    while (option != 0){
        switch (option){
            case 1 -> printProducts();
            case 2 -> printCurrentProducts();
            case 3 -> printAverageProductPrice();
            case 4 -> printCheapestProduct();
            case 5 -> printProductsAboveAPrice();
            default -> System.out.println("Invalid option entered: " + option);
        }

        //pause the program so that the user can read what we just printed to the terminal window
        System.out.println("\nPress enter key to continue...");
        input.nextLine();
        input.nextLine(); //second read is required - bug in Scanner class; a String read is ignored straight after reading an int.

        //display the main menu again
        option = mainMenu();

        //the user chose option 0, so exit the program
        System.out.println("Exiting...bye");
        System.exit(0);
    }
}
```

LCV initialised

LCV tested

LCV changed

Loop Control Variable is **option**



V2.2

```
private void runMenu(){
    int option = mainMenu();

    while (option != 0){
        switch (option){
            case 1 -> printProducts();
            case 2 -> printCurrentProducts();
            case 3 -> printAverageProductPrice();
            case 4 -> printCheapestProduct();
            case 5 -> printProductsAboveAPrice();
            default -> System.out.println("Invalid option entered: " + option);
        }

        //pause the program so that the user can read what we just printed to the terminal window
        System.out.println("\nPress enter key to continue...");
        input.nextLine();
        input.nextLine(); //second read is required - bug in Scanner class; a String read is ignored straight after reading an int.

        //display the main menu again
        option = mainMenu();
    }

    //the user chose option 0, so exit the program
    System.out.println("Exiting...bye");
    System.exit(0);
}
```

```
public static void main(String[] args) {
    Driver driver = new Driver();
    driver.processOrder();
    driver.printProducts();
    driver.printCurrentProducts();
    driver.printAverageProductPrice();
    driver.printCheapestProduct();
    driver.printProductsAboveAPrice();
}
```

V2.1

Note the methods in the switch statement are those that were in the main method in V2.1

Questions?



Thanks.

