

**Arab Academy for Science, Technology & Maritime Transport College of Computing and Information Technology**

**Course:** Object-Oriented Programming

**Lecturer:** Dr.Fahima Maghraby**,** Dr. Wael Zakaria

**Teacher Assistant:** Eng.Maiada, Eng.Hagar, Eng. Ahmed, Eng. Karim

**Sheet:** Section 10: Composition, Inheritance, ArrayList, Polymorphism

Design a program that simulates an order from a Supermarket.

The supermarket has different types of items. Each item has a name and price; however, different types of items may have specific attributes. For example: Fruits and Vegetables have a weight attribute, and the price is considered as per 1KG, while Dairy products have an expiration date and brand name. To create any item in the system, its name and price must be specified.

To make an order in the system, the customer’s data must first be entered (name, and address that consists of city, street, building, and phone number). The customer then can start adding products to the order. An order has an auto-generated ID, creation date, and a list of products along with a method that calculate the total price of the order that also contains the delivery fee. Note that if an order exceeds 500EGP then the delivery is free. Add any necessary methods to the order class that would be needed (e.g. cancel order, …etc.) and override the toString() method to return all data in the form of an invoice containing all specific details of the order. The customer has a list of all of their created orders and an order must also specify to which customer it belongs.

The main method should act as a menu of actions for the users that allows interaction with the system.

Note: start by designing the Class Diagram for the system then translate it into code

Extra self-study points:

* create an order status variable that has one of the following values (submitted, prepared, delivered, cancelled) using “Enumeration”
* update the order class to keep track of the required quantity of each added item instead of adding the item multiple times using “HashMap”
* update the stock class to use a “HashMap” and keep track of the available quantities for each item
* update the program to only add items to an order as long as a stock exists for that item