Abstract Class Challenge

In this challenge, you need to build an application, that can be a store front, for any imaginable item for sale.

Instead of the Main class we usually create, create a Store class, with a main method.

The **Store** class should:

- manage a **list of products for sale**, including displaying the product details.
- manage an order, which can just be a **list of OrderItem** objects.
- have methods to **add an item to the order**, and **print the ordered items**, so it looks like a sales receipt.

Abstract Class Challenge

Create a **ProductForSale** class that should have at least three fields: a **type**, **price**, and a **description**, and should have methods to:

- get a Sales Price, a **concrete method**, which takes a **quantity**, and **returns the quantity times the price**.
- print a Priced Line Item, **a concrete method**, which takes a **quantity**, and should **print an itemized line item** for an order, with **quantity and line item price**.
- show Details, an **abstract method**, which represents what might be **displayed** on a product page, **product type, description, price**, and so on.

Create an **OrderItem** type, that has at a minimum 2 fields, **quantity** and a **Product for Sale**.

You should create **two or three classes that extend the ProductForSale class**, that will be products in your store.

The Design

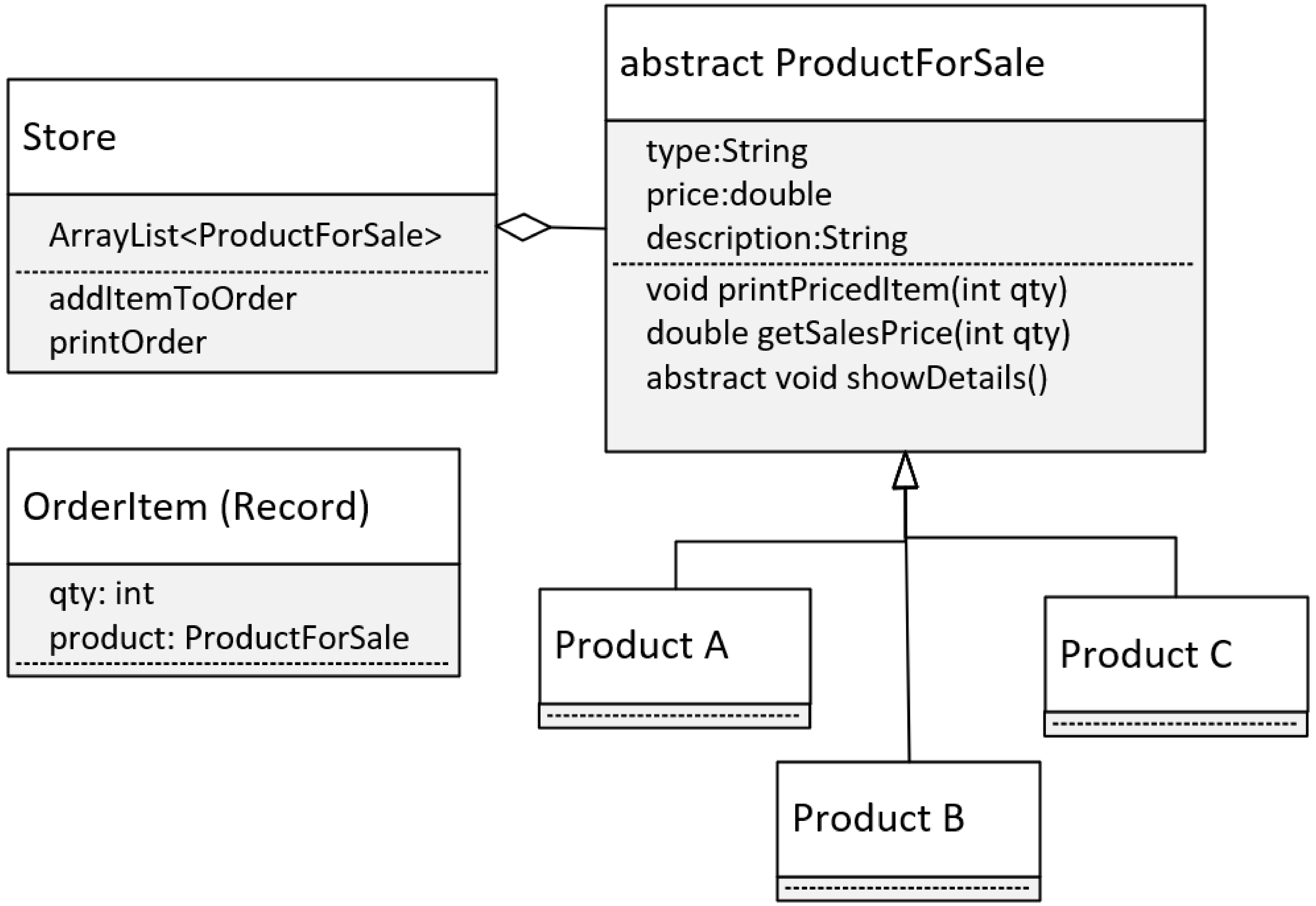
Let's look at my approach.

First, let me show you a class diagram, of what we'll be building.

This covers all the requirements we talked about.

You'll notice I'm specifying that OrderItem will be a record, and this is just to keep the code simple.

And I'm not really specifying what our store products are, we can really put anything there.


Store
ArrayList<ProductForSale>
addItemToOrder
printOrder
abstract ProductForSale
type:String
price:double
description:String
void printPricedItem(int qty)
double getSalesPrice(int qty)
abstract void showDetails()
OrderItem (Record)
qty: int
product: ProductForSale
Product A
Product B
Product C