CS12420 Assignment 2 • 2011 - 2012
AberPizza

21st March 2012 • Version 1.0

This is the individual assignment for the module. It is worth 20% of your final mark for the module. You must submit this on Blackboard by 6pm on Friday 4th May.

Overview

You are working on a new software project to develop a Till application in Java. Part of the design has been produced. You are to complete the design, implement it and test it. You have also been asked to add a user interface for the till.

Assignment Goals

By completing this assignment, you will demonstrate your ability to:

- Take an existing design and create the code that implements the design.
- Write a set of JUnit tests for the data classes that you have created.
- Enhance the design to support some additional features.
- Write a GUI for the application.
- Create Javadoc for the code that you have produced.
- Define and run tests for the user interface.

The Problem

You are to build a till application for the AberPizza store in Aberystwyth. Sales Assistants will use the application to enter order details. Each order has a customer name and a set of items, which should be selected from a list of items. Any single item, e.g. pizza, can be included multiple times in the order.

AberPizza supplies the following categories of products:

- **Pizzas** – There are different flavours and the pizzas are available in *Small*, *Medium* and *Large*.
- **Sides** – These include *garlic bread*, *fries*, *potato wedges* and *coleslaw*.
- **Drinks** – These include *cola*, *orange juice* and *lemonade*.

Other products in each of these categories may be added in future.

AberPizza runs several offers. There are offers where multiple products are bought (e.g. if 3 pizzas of the same size are purchased, a discount will be applied) and certain combinations of products will attract a discount (e.g. if a large pizza, a drink and one side dish are purchased, a discount will apply). Only one offer can apply to any one order. The application should determine which offer to apply; this should be the offer that with the biggest discount to the customer.

Use-Case Diagram

Figure 1 shows a Use-Case diagram for the till. This lists the main features that should be implemented for the till.

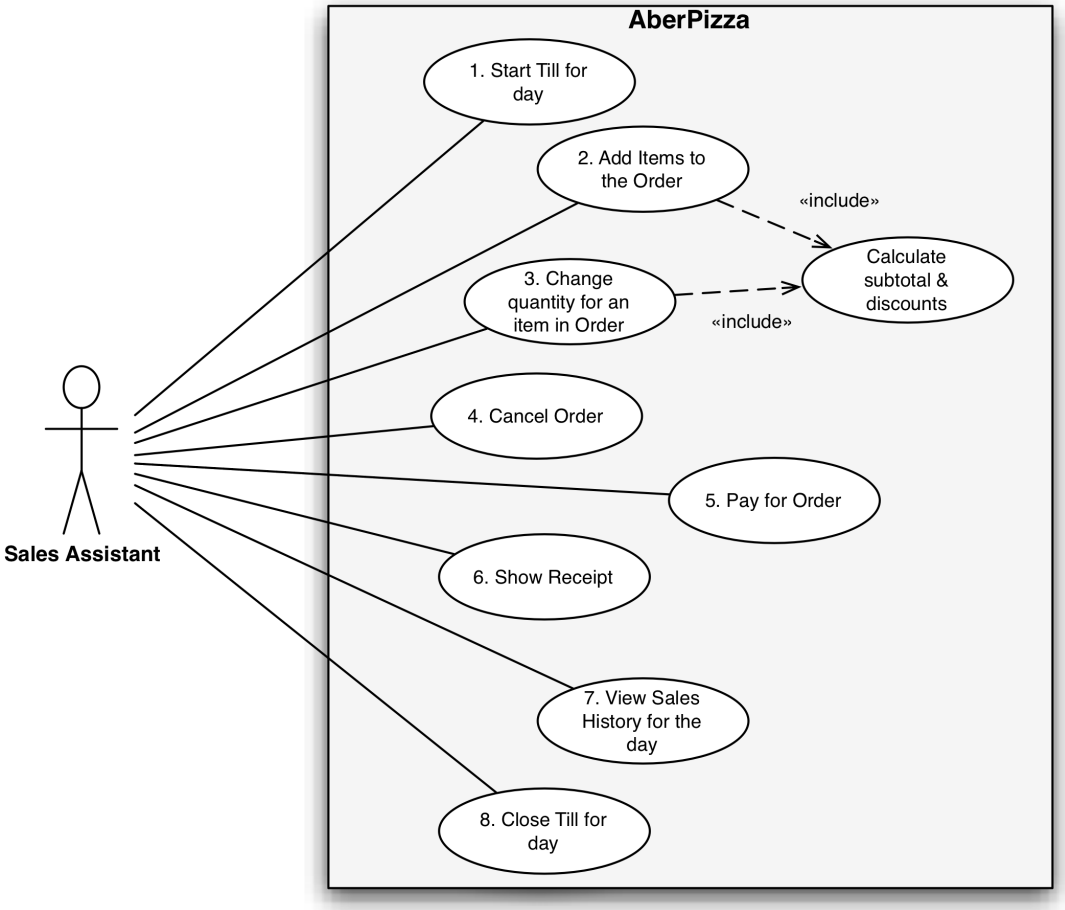


Figure 1: Use-Case Diagram for the Till

The following provides further information about the use-cases.

1. **Start Till for day** – As the program starts, it should load the previous sale records from file.
2. **Add items to the order** – A list of possible items will be shown on the screen. The user can select from the different categories of items and the select items to add. Each time an item is selected, it is added into the order of items.
3. **Change quantity for an item in order** – Once an item is in the order, it can be selected and the quantity can be changed. If the quantity is set to 0, then the item is removed from the order.
4. **Cancel Order** – At any time before the order is paid for, the user can cancel the order. This will remove the current order, clearing out any items. The application will then be ready to add items to the empty order.
5. **Pay for Order** – Paying for the order involves taking the customer's name and then entering the amount tendered. The application should confirm that this is more than the total sale price.
6. **Show Receipt** – When an order has been paid for, a receipt should be displayed. This might be in a dialog as text or as an HTML display. In the final version of the application, this would be printed out. However, printing **is not** required in this assignment.

7. **View Sales History for the day** – There should be an Admin option to produce a list of all sales for the day, formatted so that it is readable. The information in the sales history is as follows:
    a. Total number of orders
    b. A list of all orders, showing the items purchased and whether any discounts were applied.
    c. Total Amount taken for the day.
8. **Close Till for day** –When the Till is closed, the list of transactions should be saved to file ready for the next day.

In addition to the numbered use-cases, there is a shared use-case named '**Calculate subtotal & discounts**'. This is included as part of use-cases 2 and 3. This additional case is used to determine if any discounts should be applied to the order and to update the subtotal. The subtotal is the total for the items in the order, less the discount offer amount.

Existing Design

Your colleague has started a design, which is shown in Figure 2. This specifies three classes and one interface.

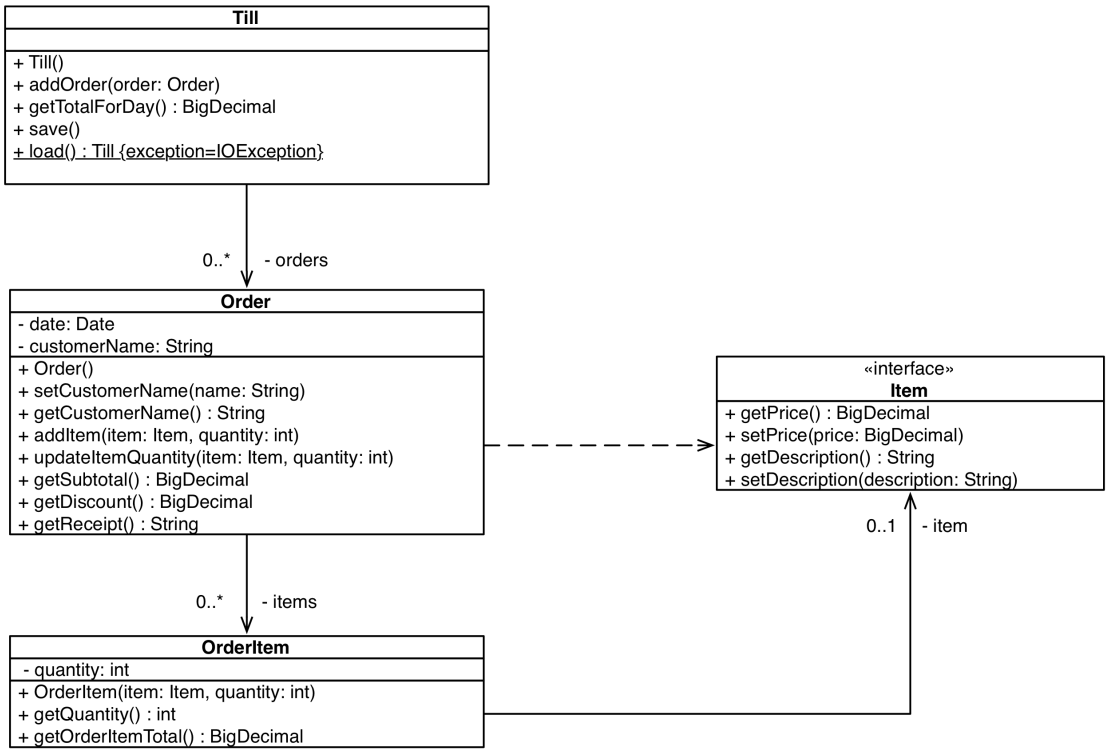


Figure 2: Initial Design

Till

This class holds a list of Order objects. The Till provides operations to save the till and load the till. You must use XML Serialisation for the save/load process.

Order

An order contains a list of OrderItem objects. There are methods to add items, use the customer's name and to get information about the subtotal and discounts for a particular order.

The order should also return a receipt. This is specified to return a string, e.g. in a text format or as HTML. It is up to you to decide what is returned and then explain your choice in your report.

OrderItem

An order item represents a link to a particular Item (which is an interface that should be used by the products in your design) and the number of the specified item. For example, you might have an item that represents a type of pizza, and the quantity is set to 2.

Item

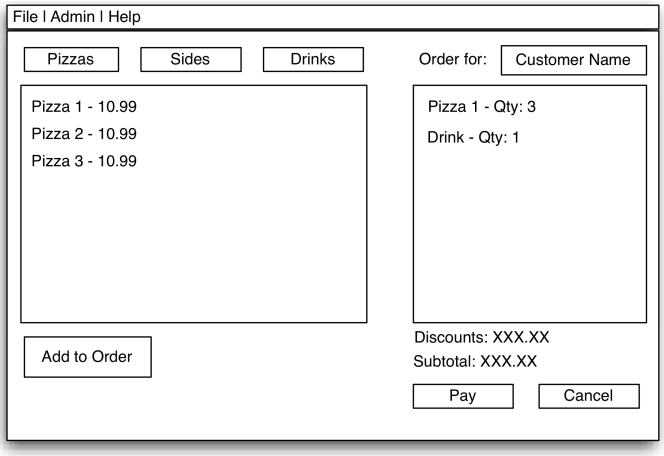
This is an interface that should be used as the basis for implementing items that are available for sale. You will need to design and implement how you will represent products that are sold. You must use this interface as part of your design.

Comments on the design

The design is incomplete. You may find places where you think there are methods missing and that you would like to change the design. Whilst the majority of this design should remain unchanged, you are permitted to make some changes, e.g. adding methods. Some changes are likely in order to support the operations for the application. As you make changes, keep information about what you have changed and why; you should discuss these changes in your report.

Graphical User Interface

You are to design and implement a GUI for the application. One possible design is shown below. You do not have to follow this design, but it might give you some ideas about the type of window you could develop.



NOTE: For this assignment, you are not allowed to use any graphical building tools. You are to write your GUI by hand using Swing code. This is deliberate for this module – we want you to understand how to create the code.

Your Task

You are to do the following:

1. Write Java code that implements the design described above.
2. Extend the design to represent the different products and add a mechanism to represent and apply offers.
3. Write JUnit tests to exercise all of your code in items 1 and 2.
4. Design and implement a graphical user interface using the Swing API.
5. Build a runnable Jar file that includes your application.
6. Test your final application.

Packages

You should organise your code into the following packages.

- The data classes, described above, should be in the package **uk.ac.aber.dcs.cs12420.aberpizza.data**.
- The JUnit tests should be in the package **uk.ac.aber.dcs.cs12420.aberpizza.tests**.
- The GUI code should be in the package **uk.ac.aber.dcs.cs12420.aberpizza.gui**.

Hand In

You need to create a ZIP file (*a .zip file, not a .rar or anything else*) that contains the following items:

1. A copy of all of your source code, e.g. your Eclipse project, and a runnable Jar file of your compiled code.
2. Javadoc for all of your code. This should be the Javadoc output generated from running the Javadoc tool.
3. A PDF document that describes the following:
    a. Your approach to implementing the design and a discussion of changes you made and the additional items you have added to the design.
    b. An explanation of the tests that you have chosen and why.
    c. A self-evaluation – say what you have found hard or easy and if anything was omitted and why.
4. A feedback form. Give yourself a realistic mark out of 100 for your work and explain why you would award this mark. **Without a feedback form, your project will not be marked.** The feedback form [1] is available online.

Submit to Blackboard

You must submit your ZIP file on Blackboard by 6pm on Friday 4th May. Use the Blackboard submission form for *Assignment 2 – AberPizza* in the Assignments section of the CS12420 module.

Note that submissions after 6pm will be marked as late. If you miss the deadline then you should still submit your work as soon as possible using Blackboard and then complete a "*Late Assignment Form*" and hand it to Neil Taylor. Your late submission will be dealt with according to the department rules specified in section 5.6 of the Student Handbook.

If your account is locked then you will not be able to submit via Blackboard. In this case you must burn everything onto a CD, confirm that the CD is readable on a different machine and hand the CD in to reception and obtain a receipt **before the deadline**.

Please upload well before the deadline, since uploads sometimes fail, servers may go down, and upload may take longer than anticipated.

IMPORTANT: Once you have uploaded your submission on Blackboard, you will be shown the *Review Submission History* screen. There will be a link for the file that you have uploaded. You should click on this link and confirm that the contents of the file match what you intended to submit. Some students accidentally submit the wrong files. **It is important that you check which files you have submitted**. If the file is not what you expected to upload, simply go through the submission process again and upload the correct file.

Effort

You are expected to spend approximately 30 hours on this project. This a general guide – some of you will send less and some of you will spend a bit more.

Marking Scheme

The marking scheme for this project is given below.

Item	Marks
Quality of the code, including variable names, indentation and Javadoc comments.	5
Design and implementation for the data classes (e.g. Till, Order, OrderItem, Item and any additional data classes that you introduce for products and offers).	25
Design and implementation of the GUI and linking it to the data classes.	20
Testing, including JUnit and test tables for the user interface.	20
WOW marks	10
Documentation. This should describe what you did to implement the code and in particular it should discuss why you chose the tests that you did. Explain what changes you made to support the additional grades. You also need to include your Javadoc output and your individual reflection.	20
TOTAL	100

Plagiarism

The department does not take cases of plagiarism in student assignments lightly, and the consequences of plagiarism by students can be severe. Please ensure that you understand what constitutes plagiarism, read the plagiarism notice from the Student Handbook; this is also available on the web [2].

You do not need to submit a paper declaration of originality form. However, you need to read the equivalent statement on the Blackboard submission page. By submitting to Blackboard, you are confirming acceptance of the statement that this is your own work.

IMPORTANT – Asking for Help

General help and answers to common questions will be put on the module discussion board on Blackboard in the forum *Assignment 2: AberPizza*. You can add and discuss issues on this forum.

If you have problems and find yourself stuck, you have several options:

1) Add questions to the forum.
2) Speak to demonstrators in your practical session.
3) Speak to advisors, available each day during term in the afternoon.
4) Contact Neil Taylor

If you are truly stuck on where to start, contact Neil Taylor.

The forum and emails will be checked over the Easter vacation, but there will be times when we are on leave and won't reply for a few days.

References

[1] Computer Science Department "Assignment Feedback Form" [Online] Available: http://www.aber.ac.uk/~dcswww/Dept/Teaching/Handbook/assign_feedback.txt (Accessed on 20th March 2012)

[2] Computer Science Department "Student Handbook" [Online] Available: http://www.aber.ac.uk/~dcswww/Dept/Teaching/Handbook/ (Accessed on 20th March 2012)