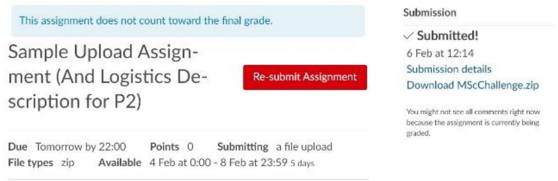
P2 OOP Programming Task (2020-21)

This task is to be completed as a new project in Eclipse (closed book, so no referring to/copying from other previously completed projects).
When you are done, put the entire project folder into a .zip file and upload to this Assignment on Canvas via the Submit Assignment button. After you submit, the button will change to "Re-submit" and a record of any files you have uploaded will be displayed in case you need to check/verify as in screenshot below.

It is your responsibility to keep an eye on your time and ensure you submit the required files on time.

PLEASE INCLUDE YOUR NAME AND STUDENT NUMBER IN THE COMMENTS AT THE TOP OF EACH JAVA FILE YOU SUBMIT



This assignment does not count toward the final grade.

Sample Upload Assignment (And Logistics Description for P2)

Due Tomorrow by 22:00 **Points** 0 **Submitting** a file upload
File types zip **Available** 4 Feb at 0:00 - 8 Feb at 23:59 5 days

Submission

✓ Submitted!
6 Feb at 12:14
Submission details
Download MScChallenge.zip

You might not see all comments right now because the assignment is currently being graded.

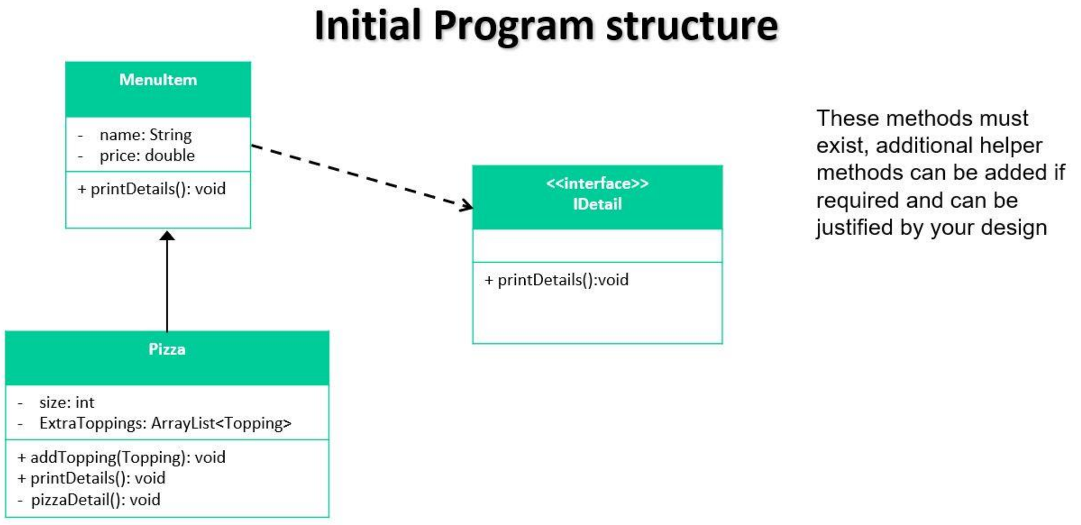
Example of upload interface after submission

Pizza Shop

The scenario to be implemented is a pizza/takeaway shop. The UML diagram below (fig 1.) and associated business rules represent the initial state of the program required, then some additional tasks and expansions are detailed in the next section.

Implement the Classes outlined in the diagram and a driver class PizzaShop.java with a main method to test and demonstrate the functionality of those classes.

Some sample console output is given as an example at the end in fig 2. Layout/formatting should be tidy, but doesn't need to match perfectly.



These methods must exist, additional helper methods can be added if required and can be justified by your design

fig 1. UML Diagram for initial program structure

P2 OOP Programming Task (2020-21)

Business Logic:

Topping - can be String or enum (Developer choice, enum preferred)

MenuItem - represents simple non customisable items with just a name and price eg Drinks/Burgers etc.

- Name must be non empty String. Letters and Spaces only and must not begin with a space. (otherwise “INVALID NAME”)
- Price: non negative, zero if invalid

Pizza - represents Pizzas which could be plain with no additional toppings or with a choice of toppings added which affects the price

- Main required constructor for Pizza(int size) - creates a plain pizza of a given size. Additional constructor(s) as developer sees fit.
- base price should equal size and name standardised e.g. size: 8, price: £8.00, name: Pizza
- Size between 8 and 16 - set to 8 if smaller value given, 16 if bigger
- Base price, then 50p for each additional Topping. Only 1 of each topping possible.

Some example toppings which could be used:

```
Extracheese

Pepperoni

Beef

Chicken

Ham

Pineapple

Peppers

Onions

Mushrooms

Olives
```

P2 OOP Programming Task (2020-21)

Section 2

Next Steps and Expansions

- Add a static method **printReceipt** to PizzaShop which receives an ArrayList of MenuItem objects and prints a summarised receipt to the console, itemising items and a total price.
  (layout should be tidy, but doesn't need to be perfect)
- Add a method **isVegetarian()** to the Pizza class which will return true if none of the Toppings are Meat based and false otherwise
- Add a static method **findVeggies** to PizzaShop which receives an ArrayList of MenuItem objects and counts the number of Vegetarian pizzas which are in that list and prints the details of each.
- Add a static method to PizzaShop which will read from an appropriately formatted csv file (samples provided) and return an ArrayList of MenuItems – consideration for invalid values in file/handling exceptions appropriately
  - Sample files: (available to download from Canvas)

Sample console output below, formatting doesn't need to match perfectly just an example:

```
****Initial Tests****


Test simple item

Burger               £5.50


Test Validation rules

INVALID NAME         £0.00


Test pizza 1

Custom Pizza     (8) £9.50

 with toppings:

 * PEPPERS

 * PINEAPPLE

 * EXTRACHEESE


Test pizza 2

Plain Pizza      (8) £8.00
```

```
Test pizza 3
Topping already on pizza, skipping
Custom Pizza     (10) £11.00
 with toppings:
 * BEEF
 * CHICKEN

****Expansions and searches Tests****

Test receipt method
--------------
ORDER DETAILS
--------------
Burger               £5.50
Custom Pizza     (8) £9.50
 with toppings:
 * PEPPERS
 * PINEAPPLE
 * EXTRACHEESE
Plain Pizza      (8) £8.00
Custom Pizza     (8) £9.50
 with toppings:
 * MUSHROOMS
 * OLIVES
 * EXTRACHEESE
--------------
Total cost: £32.50
--------------

Test veggie search method
Veggie 1)
Custom Pizza     (8) £9.50
 with toppings:
 * PEPPERS
 * PINEAPPLE
 * EXTRACHEESE


Veggie 2)
Plain Pizza      (8) £8.00


Veggie 3)
Custom Pizza     (8) £9.50
 with toppings:
 * MUSHROOMS
 * OLIVES
 * EXTRACHEESE
A total of 3 Veggie Pizzas in this list
Test file read method (by printing receipt)


--------------
ORDER DETAILS
--------------
Burger               £2.50
Custom Pizza     (12) £14.00
 with toppings:
 * CHICKEN
 * EXTRACHEESE
 * PEPPERS
 * BEEF
Coke                 £0.80
Chicken Nuggets      £1.50
Custom Pizza     (14) £14.50
 with toppings:
 * EXTRACHEESE
Custom Pizza     (8) £9.50
 with toppings:
 * EXTRACHEESE
 * HAM
 * PINEAPPLE
Cheeseburger         £2.95
Coke                 £0.80
Plain Pizza      (14) £14.00
Chips                £2.50
Chips                £2.50
Plain Pizza      (10) £10.00
Custom Pizza     (13) £14.00
 with toppings:
 * PEPPERS
 * ONIONS
Plain Pizza      (15) £15.00
Burger               £2.50
--------------
Total cost: £107.05
--------------
```

fig 2. Sample console output