**Object Oriented Development**

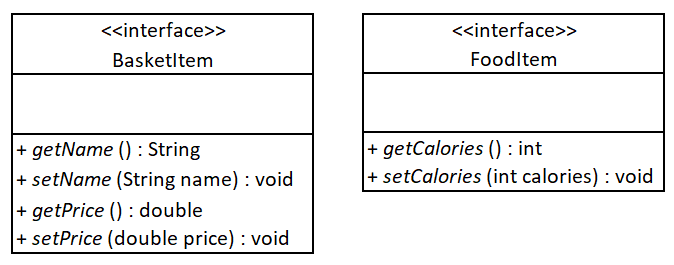
Module 10 : Inheritance - Interfaces

© FDM Group Ltd 2020. All Rights Reserved.

Any unauthorised reproduction or distribution in part  
or in whole will constitute an infringement of copyright.

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Author | Comments |
| 1.0 | 21 / 10 / 20 | Nick Lawton | First draft |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

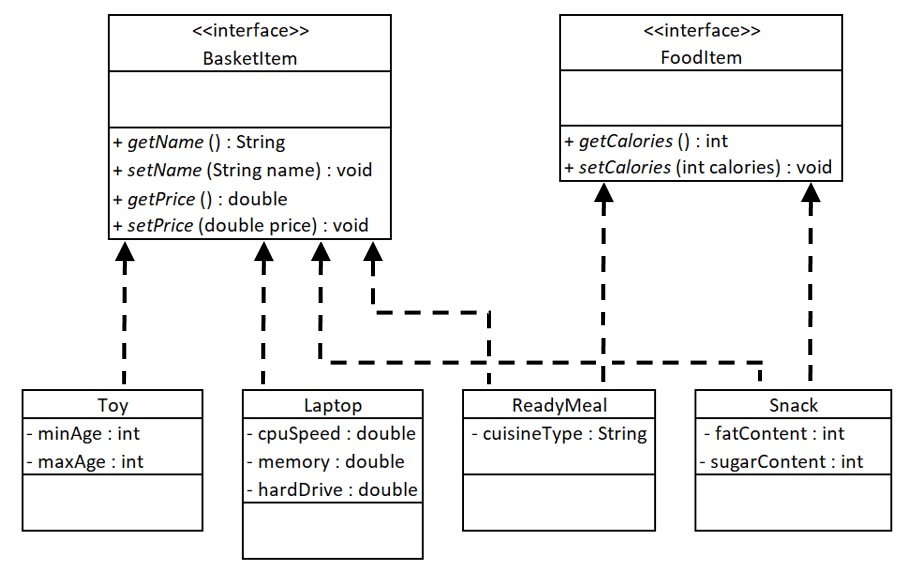
## Part 1 – Creating interfaces



Create a package called com.fdmgroup.interfacesExercise.

Inside the interfacesExercise package, create the BasketItem and FootItem interfaces shown in the UML above. Ensure that all methods are abstract.

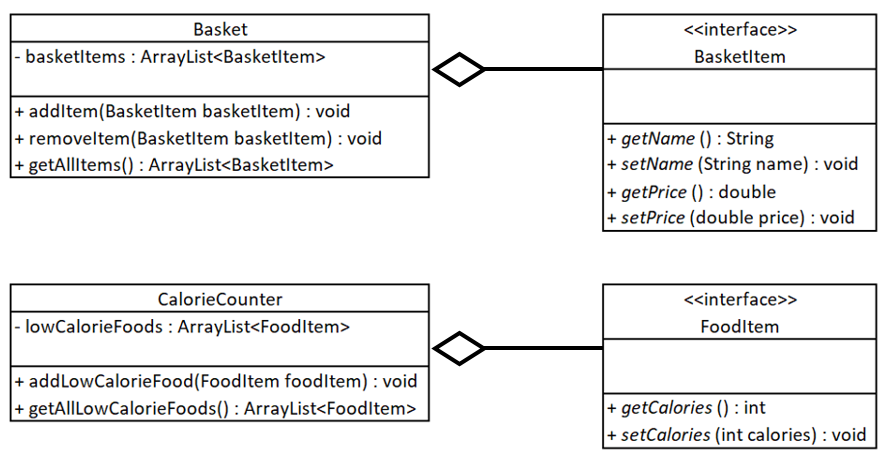
## Part 2 – Creating classes which implement the interfaces



Create the 4 classes in the UML above, implementing the interface(s) shown in the diagram.

Create private variables in each class to match the getter and setter methods inherited from the interface(s). Write code in each class to implement all the methods inherited from the interface(s).

## Part 3 – Creating classes which depend on the interfaces



Create the Basket and CalorieCounter classes as shown in the UML above.

Basket:

* The basketItems ArrayList object should be instantiated within the Basket class. It should not have a setter method.
* The addItem() method should add the BasketItem argument to the basketItems ArrayList.
* The removeItem() method should remove the BasketItem argument from the basketItems ArrayList.
* The getAllItems() method should return the basketItems ArrayList.

CalorieCounter

* The lowCalorieFoods ArrayList object should be instantiated within the CalorieCounter class. It should not have a setter method.
* The addLowCalorieFood() method should check if the FoodItem argument has 500 calories or less. If it does then it should be added to the lowCalorieFoods ArrayList.
* The getAllLowCalorieFoods() method should return the lowCalorieFoods ArrayList.

## Part 4 – Creating objects and calling methods

Create a class called Runner with a main method. All the code in this section will be written in the main method.

### Creating objects

* Create one Toy object and set all of its attributes to values of your choice.
* Create one Laptop object and set all of its attributes to values of your choice.
* Create two Snack objects, one should have more than 500 calories, the other should have less than 500 calories. Set all other attributes to values of your choice.
* Create two ReadyMeal objects, one should have more than 500 calories, the other should have less than 500 calories. Set all other attributes to values of your choice.
* Create one Basket object and one CalorieCounter object.

### Calling methods

Basket:

* Call addItem() once for each of the 6 BasketItem objects.
* Call getAllItems() and loop through the ArrayList that it returns, printing out the name of each item. Verify that the names of all 6 items are present.
* Call removeItem() for one of the BasketItem objects.
* Call getAllItems() and loop through the ArrayList that it returns, printing out the name of each item. Verify that there are now only 5 names present and that the name of the item you removed is no longer there.

CalorieCounter

* Call addLowCalorieFood() once for each of the 4 FoodItem objects.
* Try adding a Toy object and a Laptop object to addLowCalorieFood(). Verify that this produces a compile error.
* Call getAllLowCalorieFoods() and loop through the ArrayList that it returns, printing out the number of calories for each food. Verify that there are only two items present and that both have no more than 500 calories.