CS1002 — OBJECT-ORIENTED PROGRAMMING

CS1002 CLASS TEST 2—INSTRUCTIONS

Deadline: Friday 10th November 2023 at 11:00

Weighting: 20% of module grade

Location: John Honey Teaching Lab (JH110)

MMS Slot: CS1002 Class Test 2 > TEST2-1000

> **The class test will take place under exam conditions, in the John Honey lab. You cannot take this test remotely. You must find a workstation, and place your student ID card face up so that your ID number is clearly visible to the invigilators. You must not talk or communicate with anyone else during the test. Phones, tablets and other personal electronic devices cannot be used and cannot be on your desk during the test. You are not permitted to leave the test before 10:55.**
>
> **The test will start at 10:05. You must submit your answers before the deadline (11:00).**

INSTRUCTIONS

PERMITTED RESOURCES

You are permitted to use VS Code (to write and run your code), Moodle (to download the specification and starter code), and MySaint/MMS (to upload your solution). You can use VS Code extensions (**excluding any AI tools**) if you wish, but you don't have to.

> If you have installed any VS Code extensions you must check, before you begin the test, for any AI-related extensions, including, but not limited to, *IntelliCode*. Any such extensions must be disabled or uninstalled for the duration of the test. Failure to do so may be treated as potential academic misconduct.

You may access studres to refer to the CS1002 course material. You may also access your own CS network home directory to view your solutions to CS1002 exercises.

You are **not permitted to use any other resources**. If you open any other programs or windows (e.g. email, Teams, websites, etc.) you will be asked to leave and procedures for breach of Good Academic Practice will be initiated. This could result in a mark of zero for the class test, or failure of the module overall.

TASK

In this test, you are given some incomplete starter code (the classes *ClassTest2*, *Shop* and *ShoppingCentre*), which initially won't compile. You need to first add enough code to remove the

compilation errors, and then add further code according to the instructions in the code comments. You will need to add new code to all three classes. **For *Shop* and *ShoppingCentre* you will need to define additional attributes, methods and constructors where necessary**.

The scenario involves storing and retrieving information about some shops (e.g. Tesco, Morrisons, Co-op etc) within a shopping centre, and for each shop, the names of some items that they sell (e.g. bread, cheese, carrots, etc). Your code should allow up to (and including) five shops to be stored, with up to (and including) five items for each shop.

The body of the method *recordAndPrintShops* in *ClassTest2* is already provided. It reads in a series of shop names from the user, and calls a method in *ShoppingCentre* to record a shop object for each one. Then it calls another method to print out the names of the shops. You will need to define any methods that are not already present.

The correct operation of *recordAndPrintShops* is tested in the *stacscheck* tests *test01* and *test02*. You may find it useful to look at the expected output for each test, which is defined in the corresponding `.out` file within the directory *Tests/public*. For example, *test01.in* and *test01.out* show that if the user types in `1` in order to select *recordAndPrintShops*, and then enters `Tesco`, `Fortnum & Mason` and `Morrison`, followed by `done`, then the output should be those three shop names in the order that they were entered.

At any point during the class test you can check your progress by running *stacscheck* in the terminal, within the *ClassTest2* directory:

```
stacscheck Tests
```

It's suggested that you focus on each test in turn. In order to pass *test01* and *test02* you will need to add code to *Shop* and *ShoppingCentre*. For the other tests you need to add code to all classes.

The *findItemsForShop* method needs to create a series of shops as in the previous method. In addition, for each shop, it needs to read a series of items from the user, and store them in the shop object. Finally, it should prompt the user for the name of a particular shop, retrieve the items sold by that shop, and print them out.

The *findShopsSellingItem* method should create shops and items in the same way as the previous method, and then prompt the user for the name of a particular item. It should then find all the shops that sell that item, and print out their names.

You can copy and paste code between the methods in *ClassTest2* if you find it helpful. If you have time left after you have got everything working, you could try to factor out any repeated code into some extra methods, to reduce the amount of code duplication.

> Each time you manage to get an additional test passing, make a copy of your current *.java* files in a separate directory. Then if you are half-way through changing your code when the class test ends and it doesn't currently compile, you can go back to a previous version before submitting.

MARKING

Your mark will be determined largely by the results of the automated checker, including both public tests available to you during the test, and private tests. Any failing tests will be investigated by a human marker to ensure appropriate partial marks are awarded. Your tutor will check the style of the code, including layout and naming conventions, which will be a small portion of the mark for this test.

SUBMISSION

Submit your class test solution to MMS as a **ZIP** file to the TEST2-1000 slot in CS1002 Class Test 2. As always, you should receive an automatic email receipt, which you should keep since it proves that you uploaded the file.

Once you have uploaded your solution, don't forget to immediately download it again and **make sure that you didn't accidentally upload the starter code**.

Author: Prof Graham Kirby

Module: CS1002 – Class Test 2