Software Development 1, Coursework 2

This is individual assessed coursework. You are allowed to discuss this assessment with other students, but you should not copy their code, and you should not share your own code with other students. We will carry out plagiarism checks on submissions.

Objectives

This coursework contributes 15% of your overall mark for the course. It assesses your understanding of the learning materials from the first half of the course. In addition to the topics already assessed in Coursework 1, this coursework also assesses the following topics covered in Weeks 4 and 5:

- Creating, storing to, accessing, and iterating through arrays
- Using loops to implement repetitive behaviour
- Nesting conditional execution within loops, or vice versa

It is recommended that you work through both the recorded material and the tutorial exercises from these weeks before attempting the coursework, since these help to build the understanding you need to complete the coursework.

Submission

The **deadline** for completing this coursework is **Monday 24th October (Week 7)**. Make sure you do these two things:

1) **Commit your work to GitLab** before this deadline. If you don't do this, a late penalty may be applied. There are instructions on using GitLab in the Week 1 familiarisation tutorial.
2) **For students in Edinburgh, attend a lab session and ask someone to mark your work**. After they've done this, they will give you feedback on your work. You may get your work marked before the deadline or in the week following the deadline without a penalty, so long as you committed it to GitLab before the deadline.

Do This First: Fork and Import the GitLab Project

You will be adding your code to an existing project in GitLab. The first thing you need to do is fork this project, so that you'll be working on your own copy, and import it into Eclipse:

1. Visit https://gitlab-student.macs.hw.ac.uk/F27SA_2022-23/f27sa_2022-23_coursework-2
2. Fork the project
3. Import your forked project into Eclipse

! ***Don't forget to fork the project before you import it***
if you don't do this, you won't be able to save your changes back to Gitlab.

If you don't know how to do any of this, then make sure you first work through the Week 1 tutorial, which will take you through the process step by step. If you're having trouble getting it to work, then come to a timetabled lab session and talk to a lab helper, well before the submission deadline.

(continues on next page…)

The Behaviour of Your Programme

The GitLab project contains a single class, **MushroomDatabase.java**, where you are going to add your code. You are required to write a very simple implementation of a database, which will read data records from the user, store them in arrays, and then allow the user to print out selected records according to some criteria. Since it's autumn in the UK, it is themed around mushrooms.

Here's an example of what your program's behaviour could look like (user input shown in bold):

```
Welcome to Mushroom Database!
How many records to you wish to input?
3

Enter details for record 1
Name:
InkyGungeCap
Colour:
green
Edibility (y/n):
y

Enter details for record 2
Name:
RedRimRoll
Colour:
red
Edibility (y/n):
n

Enter details for record 3
Name:
SmellySulphurCap
Colour:
red
Edibility (y/n):
n

List (a)ll, search by (c)olour or (e)dibility, or (q)uit?
a
These are all the mushrooms in the database:
InkyGungeCap
RedRimRoll
SmellySulphurCap

List (a)ll, search by (c)olour or (e)dibility, or (q)uit?
c
Which colour?
red
These are all the red mushrooms in the database:
RedRimRoll
SmellySulphurCap
```

(continues on next page…)

```
List (a)ll, search by (c)olour or (e)dibility, or (q)uit?
e
Search for (e)dible or (n)on-edible?
e
These are all the edible mushrooms in the database:
InkyGungeCap

List (a)ll, search by (c)olour or (e)dibility, or (q)uit?
q
Thank you for using Mushroom Database!
```

Specifically, you should write code so that:

- The user specifies how many mushroom data records are to be read in.
- The name, colour and edibility of each type of mushroom should then be sequentially read in from the user and stored in **arrays**.
  - This data should be stored in three separate arrays: one for names, one for colours, and one for edibilities, i.e. the first element of the names array should be the name of the first mushroom, the first element of the colours array should be the colour of the first mushroom, the first element of the edibilities array should be the edibility of the first mushroom, the second element of the names array should be the name of the second mushroom, and so on.
  - To keep things simple, it is okay to assume that names will be entered as single words, as in the example. This will allow you to use Scanner's `next()` method rather than `nextLine()`, since the latter can be a bit troublesome.
- The edibility array should be a **boolean array**, since this is more efficient than storing binary values as strings or characters.
  - That is, "edible" should be stored as true, and "not edible" should be stored as false.
- Once the user has finished entering the data, they should be given the option to:
  - List all the entries in the database. You can just print the names, as in the example.
  - List all the entries that match a particular colour, as specified by the user.
  - List all the entries that match a particular edibility, as specified by the user.
  - Quit, at which point the program should display a farewell message and exit.
- The program should keep asking the user what they want to do, until they choose to exit.
  - You will need an outer loop to implement this behaviour.
  - Note that your program will automatically exit when execution reaches the end of the main method. You should not use `System.exit()` to achieve this.

As usual, it is recommended that you commit your changes to GitLab every time you get something working. If you later break it, this will make it easy for you to restore the earlier working version. Once you've finished, **make sure you commit your code to GitLab** before you get it marked. If you have not been able to finish everything, please commit what you have been able to finish.

(continues on next page…)

Marks

You will get a mark out of 15 for your work. Here is a guide to how your work will be marked, though the final mark is up to the person marking your work, and will reflect the understanding of the course materials shown in your work**:**

- 5 marks for correctly reading data from the user and storing it in arrays
- 5 marks for correctly accessing the arrays and displaying the correct data
- 2 marks for correctly implementing the user interface
- 2 marks for programming style, including whether you've made good design choices, how your code is formatted and commented, and whether it follows Java's naming conventions
- 1 mark for using GitLab correctly, i.e. forking, checking out the code stub, and committing

Late submissions will be marked according to the university's late submissions policy, i.e. a 30% deduction if submitted within 5 working days of the original deadline, and no mark after that.

If you have mitigating circumstances (e.g. illness), please submit a mitigating circumstances application, as described at:

https://www.hw.ac.uk/students/studies/examinations/mitigating-circumstances.htm