COMPGC01 Introductory Programming
Coursework
Due date: 19 December, 2016, 23:55
Feedback returned: One month after demos running in weeks 21-24
Moodle Submission Only

COMPGC01 will assess your ability to demonstrate your programming knowledge, purely in Java. It does not require a fully completed application to get a successful mark. However, the more completed it is, the higher your marks will be as you will see from the mark scheme below.

You have two project options on this COMPGC01 coursework to choose from:

1. Work individually on a restaurant billing management system.
2. Work in pairs on a robot simulator application.

Option 1 - Restaurant Billing Management System (individual project)

You are required to design and implement a restaurant billing management system with the following minimum requirements:

1. The management system should be implemented as a JavaFX GUI application using buttons, tables, listeners, etc.

2. A restaurant employee should be able to login to the restaurant management system using a username/password pair.

3. The system should display a graphical representation of the tables with their numeric labels and approximate position within the restaurant.

4. Register a new order with minimal information such as table number, time of order, list of ordered items, total amount of order, special requests, comments, etc, by first clicking on the graphical representation of a table.

5. Clicking on the graphical representation of a table again allows editing its current order (add, modify, delete, etc).

6. Delete an exiting order. Your program should ask for confirmation before deleting an order.

7. Search the list of orders based on any of the stored fields, i.e., table number, dates and times intervals, ordered food, etc. and produce a list of corresponding orders.

8. The restaurant manager has a special account that allows him/her manage employees accounts (add, delete accounts) and display their activities log. Edit menus, etc.

9. The restaurant manager can also export a list of selected orders as a comma separated file or any other format.

10. The restaurant manager can also import a comma separated file (or the format used for export) containing orders into the system. The new orders data will be appended to the existing ones already stored in the restaurant management system.

OR

Option 2 - A wheeled robot simulator (group project)

In a group of ***two students***, you are required to design and implement a wheeled robot simulator with the following minimum requirements:

1. The application should be implemented as a JavaFX GUI application using buttons, listeners, etc.

2. The robot is represented by a rectangle, or an image.

3. The robot can a) move forward or backward with a given speed b) rotate left/right with a given angle.

4. A user can control the robot through a set of GUI commands such as buttons and text fields, etc, or keyboard keys.

5. The robot can move in a maze that can be represented by a set of connected lines.

6. A text or audio alert is triggered if a move would cause the robot to collide with one of the walls (lines representing a wall).

7. The robot has an initial battery charge that decreases each time the robot moves one unit of distance. The robot stops if battery charge is equal to zero.

8. A text or audio alert is triggered when the robot charge is less than 10% of the maximal charge.

9. A text (java text area for example) is updated live with the coordinates of the robot during movement; it's angle, distance crossed and battery level.

10. Record all commands executed by the robot in a text file with a timestamp.

11. Read commands from a text file and execute them by the robot at once.

12. Study differential steering of wheeled robots and provide an implementation. Here is a starting point: http://rossum.sourceforge.net/papers/DiffSteer/

13. The parameters corresponding to the robot geometry such as the dimension of the robot, the distance between the two wheels, the wheels radius, etc are stored in an XML file and loaded when the application starts. The implementation provided in requirement (12) should be adapted accordingly.

Deliverables

The grade for your GC01 coursework will depend on the quality and correctness of your programming implementation. You are required to submit a single ZIP file on moodle containing:

1. The source code files of your project, including the Eclipse project workspace (should be able to compile as is)
2. The class-generated files of your project in JAR format.
3. A text file list of the features implemented.
4. Javadoc appendix.
5. A **link** to your youtube video demonstrating all the features of your application. You can add voice-over or text comments.
6. (***Optional***) tests folder for any other demos (compiled and source code) that stand alone in demonstrating features of your system, perhaps you had a number of prototypes as you learnt.

Mark scheme

In both options, you will be assessed clearly on the following, which must be shown in a useful context. *Please note that all non-optional requirements must be implemented for a 70%:*

1. All requested features implemented
2. Has appropriate Java objects and demonstrates inheritance.
3. Makes use of Arrays, Arraylists or other Java.Util collections where appropriate.
4. Has a fully implemented Java Graphical Interface.
5. Java exceptions are used and code held in Java Packages.
6. JavaDoc comments generated and a JavaDoc folder submitted.

Distinction marks

The resulting application can include any additional feature you might think your software should have and not listed above. Your final mark will therefore depend on the innovative aspects you add to your software.

Notes

- This coursework is compulsory.
- If you include the use of 3rd party libraries in your solution (which is welcomed so long as it is cited in the comments in that Java source file) ensure any cited works are free and open for reuse.

Tips

- Start this coursework immediately, so not to get behind with other coursework deadlines.
- Revise eclipse strategies for creating packages, compiling, debugging methods, add breakpoints, how to load existing Eclipse projects and how to import other libraries.
- Refer to the 'useful links' section below for some additional tips.

Reminder

Plagiarism of any kind, on the applications and on your content prepared will not be tolerated! All of the software developed must be your own works. Any code examples used from online sources and tutorials must be CITED. Failure to do so will account to Plagiarism, which will be dealt with under the appropriate Examination Boards.

You will learn along the way how best to test it with small test apps. You may keep these in a separate folder called tests if you wish to submit it with your main Zip file as supporting evidence of your project's capabilities. This will greatly help you in future programming courses.

Useful links

1. ***Up and Running with Eclipse*** with Charles Kelly. Available from the UCL Lynda website https://www.ucl.ac.uk/lynda

2. ***User Interface Design For Programmers*** by Joel Spolsky. A very long article on how to design intuitive user interfaces. ***Tip***: The last paragraph summarizes the main tips: http://www.joelonsoftware.com/uibook/fog0000000249.html

3. ***Object-oriented design principles and the 5 ways of creating SOLID applications*** by Kaur Matas. http://zeroturnaround.com/rebellabs/object-oriented-design-principles-and-the-5-ways-of-creating-solid-applications/

4. **Late submissions**: Please consult your academic manual.

~~~ End ~~~
~~~