Department of Computer Science
Assessed Coursework Set Front Page

Module code: CS2JA16

Lecturer responsible: Prof Richard Mitchell

Coursework description: Major Coursework #1

Work to be submitted on-line via Blackboard by 12:00 noon on: 18/01/18

Work will be marked and returned by: 08/02/18

NOTES:

This coursework should be submitted on-line through Blackboard Learn.

By submitting this work you are certifying that it is all your own work and that use of material from other sources has been properly and fully acknowledged in the text. You are also confirming that you have read and understood the University's Statement of Academic Misconduct, available on the University web-pages.

If your work is submitted after the deadline, *10%* of the maximum possible mark will be deducted for *each* working day (or part of) it is late. A mark of zero will be awarded if your work is submitted more than 5 working days late. You are strongly recommended to submit work by the deadline as a late submission on one piece of work can impact on other work. If you believe that you have a valid reason for failing to meet a deadline then you should complete an Extenuating Circumstances form and submit it to the Student Information Centre *before* the deadline, or as soon as is practicable afterwards, explaining why.

MARKING CRITERIA

First Class (>= 70%)	Strong technical knowledge and skill shown through development, proving a strong grasp of object orientation and advanced programming. Report is well structured and fluently written. Design choices are validated in the report, and the work is showing study beyond the standard material.
Upper Second (60..70)	A solid grasp of the subject with a good selection of advanced programming methods. The report is well written, and validates design choices. May show some elements of creativity and originality, and makes use of existing literature to validate choices.
Lower Second (50..60)	A reasonable range of grasp of the subject, with few technical errors and written in plain English. On topic, relevant, and relatively well organised.
Third (40..50)	Evidence of appropriate study showing success in progress towards providing a solution with most technical content correct. The work relies on simple examples or uses methods inconsistently.
Pass (30..40)	Shows some evidence of study, but may be largely unfinished, flawed, or irrelevant, whilst showing some attempt to present a coherent solution.

Major Coursework #1

In this coursework you are to implement a simulator of different types of robot moving around an arena in which there are other robots, lights, chargers and other obstacles, which is displayed in a graphical user interface using JavaFX. The user should be able to define the size of the arena and the contents.

Each student may determine the richness and complexity of the simulation and use creativity: remember that richness impacts positively on the final mark.

The coursework you write should be influenced by the work done in Weeks 3, 4 and 5, to produce robots moving around an arena demonstrated in a console interface. That work, demonstrated in Week 6, contributes 10% of the marks for this work. However, you should start the GUI work afresh, in a new package, also using ideas and methods gleaned from the practicals in Weeks 7 and 8.

You should carefully design your work and use the timetabled sessions to implement your project, where help is available. You can work on the project outside that time if needed.

Graphical User Interface

A Graphical User Interface (GUI) based on JavaFX must be provided. You can use the Java console (stdin/stdout) only for displaying debugging information. The GUI should have menus, a toolbar with buttons and an information panel – as set out below. ***IMPORTANT***: A GUI based on GUI builder tools will not be accepted: the GUI must be hand-coded by using the JavaFX API directly.

File Handling and Configuration

The application should allow the user to save and load simulation configurations from files. A configuration is defined as the set of parameters required to set up and start a simulation. By default, the simulator should load from a configuration file, but if one is not there then a default arena with at least three objects in it should be provided.

Application Menu

The application should be provided with a suitable menu, which allows the arena to be configured, saved and edited, that allows the simulator to run, and to provide help. You can make design choices about the menu items, as long as you describe these choices in the design section of the report.

Toolbar

Also required is a toolbar with buttons to control the simulation (e.g., start, pause, add a robot etc.).

Information Panel

As the simulation runs, information should be provided on the state of each item, such as its position, amount of energy, etc

Classes and Inheritance

The code should make use of an abstract class for an item in the arena, which is inherited directly or indirectly by all types of robot, light, etc. Pay special attention to your use of inheritance, and use access modifiers correctly. You should also have a class for the GUI and one for the arena.

Animation

You can use any method, as long as you use JavaFX, to achieve the animation.

Comments and documentation

Your code should be professionally laid out and commented using Javadoc style comments.

Demonstration and Submission

Final Demonstration: You are required to demonstrate the final application and to get a demo mark sheet signed by a marker during your practical session in the **Week 2** of the **Spring Term**. Missing the demonstration will impact your final mark negatively. If you have an officially approved extenuating circumstance then contact Richard Mitchell by e-mail (r.j.mitchell@reading.ac.uk) to set up an alternative demonstration timeslot.

Electronic submission:

1. You are required to submit an electronic **report in docx / pdf format** on Blackboard by the specified deadline. The report (max 10 pages including appendices/figures of A4, 12 pt font) should include:
   a. A brief abstract. A short introduction.
   b. A description of the OOP design of your application (list/explanation of classes, UML diagrams, data design etc.) with a critical analysis of the design used. (Remember to use appropriate software engineering methods)
   c. A short informal presentation of the final application (including screenshots of GUI) and a brief user manual,
   d. Tests, discussion and short conclusions with your personal reflection on this project.
2. You are required to submit an electronic copy of your code (**runnable jar archive including source code**) in Blackboard by the deadline specified in the header.

   Note, using Eclipse you can easily export your project as jar archive. Please make sure that it contains the source code (**mandatory**) and can be executed (**mandatory**). You can find more information on jar archives here:
   http://docs.oracle.com/javase/tutorial/deployment/jar/build.html

   The jar archive must contain the source code and must be able to be run using the command:

   ```
   java -jar studentName_ALS.jar
   ```

   Include also a configuration file.

   Test that your JAR works before you submit it.

   The JAR file should include a top-level folder named “javadoc” containing the Javadoc documentation as a website.

Development

In the lab sessions in weeks 9, 10 and 11, and in your own time as appropriate, you should design and then write the relevant code. Your design should build on concepts introduced in the console based simulator in weeks 3-5, and your work on GUIs in weeks 7 and 8. The following suggests a framework for this, but you are free to follow another path as long as the code you develop meets the specification.

Hint : it is suggested that all items in the arena are basically circular, with potentially extra facilities (eg wheels and sensors for the robot). For moving objects, have the speed it is travelling and the angle it is oriented (hence direction it travels).

You can then borrow some of the ideas in the ball examples when it comes to checking if they are hitting another item.

Please watch the video associated with this document, which has some basic ideas on the Robot GUI, which covers the abstract class and drawing, whisker and beam sensors, and the class for a Line which you are welcome to use or adapt.

You are advised to look again at the robot simulations used in Begin Robotics. These are at:

https://www.reading.ac.uk/UnivRead/vr/OpenOnlineCourses/Files/Simulation2/BeginRoboticsSimulationindex2.pdf

Suggested List of Things to do in Week 9

Create a new package for the Robot Simulator in a GUI.

Create a new class for the GUI, based on that for the Solar System.

Create a new abstract class for an item in that arena, with a unique identifying number, which has a position, size and colour.

Create a new class Robot, which inherits from your abstract class which travels at a given angle.

Create a new class for the Robot arena, which has an array list for items in the arena.

Define relevant variables and write the appropriate methods in these classes so that at least one robot can be added to the arena and shown there.

A simple behaviour for a basic robot, similar to that in the robot console example, is that when it detects that it cannot move to the next location, it turns through 90 degrees. Add this behaviour to the Robot class, and add an option to the GUI to get the robots to move around.

Things to do in Weeks 10 and 11

Robots should be added which have different sensors. One such sensor is a ‘whisker’ which can report if it detects another object or a wall of the arena. A robot with such sensors on the left and right could be programmed to move around the arena avoiding obstacles – as per Begin Robotics week 2.

An echo-locating sensor could be added, which can be implemented as a beam which detects an item or wall if it is within the beam. This can be done by simulating the beam as a series of whiskers.

Light objects could be added to the arena, together with robots with sensors that detect lights, and steer towards them. A light is detected if it is within a ‘beam’ emanating from the robot.

Look back at Begin Robotics for other robot behaviours.

Think carefully about how lights and different robots fit into the class hierarchy.

Coursework #1: FINAL DEMONSTRATION MARK FORM

Print this out, fill in your assessment in first column **BEFORE** the demonstration and be set up ready to show and explain your code to the marker

Your full name (PRINT):		**MARK :**

	Students own assessment	Marker's observations during Demo	Mark
1.	Overall **OOP design**, API and code style: following Java code conventions: ☐ variable and class names ☐ method names ☐ indentation rules ☐ using inline developer comments, ☐ using Javadocs comments, ☐ abstract class ☐ good hierarchy for items in arena Overall OOP approach **must be described in the report under the OOP design** including abstraction, encapsulation, and information hiding choices. Appropriate use of inheritance, access modifiers for classes, attributes and methods (Information Hiding).	Number of Classes: ☐ ☐ Inheritance used Access modifiers check: ☐ Correct ☐ Less than 5 incorrect ☐ More than 5 incorrect ☐ Static variables and methods used correctly	0-25
2.	JavaFX **GUI** design and implementation: usability, robustness, quality, user-friendliness. ☐ Crashes ☐ Feedback to user instead of crashing or recover ☐ Runs smoothly without interruptions ☐ Responsive ☐ Lagging functionality **This must be described in the report under the informal presentation**	Design quality: ☐ Professional looking ☐ Understandable ☐ Lacking clarity ☐ Toolbar buttons ☐ Menu ☐ Animation view ☐ File handling	0-10
3.	**Animation**: and control ☐ Animation attempted ☐ Animation works ☐ Start/stop ☐ Pause/restart ☐ Menu control ☐ Toolbar buttons	Comments	0-10
4.	**Artificial World**: does it support different entities ☐ Robot which turns 90 degrees on detecting item/wall ☐ Robot with whisker ☐ Light ☐ Obstacle ☐ Other – specify: ☐ Other – specify: ☐ Other – specify: ☐ Other – specify: **Design must be described in the report**	Total number of different item types in arena: ☐ Comments:	0-20
5.	Quality of the submitted **report**. Tests and discussion in report Executable jar archive submitted electronically with embedded javadoc.	**Evaluation based purely on quality of submitted work.**	0-25

Signed and dated (student): **Signed (marker):**

Receipt CS2JA16 CW 1 Demo	Student Name	Mark	Marker Signature