COEN 275 Object Oriented Analysis & Design Winter 2016

Individual Project Due Date: 21st Feb.

Description:

Analyze, design, and implement an automated gardening system. You will only need to turn in your Java implementation and user guide/manual. However, if you do not analyze the requirements and design your system prior to starting your implementation you will fail and receive a very low score!

Requirement Analysis:

Find out what all is needed for an automated gardening system.

Resources you should utilize:

- Gardening books (Go to a library if you have no gardening books)
- Talking to a gardener
- The internet

Design:

Design your system. There is no need to use UML for this project. In fact my hope is that the difficulty in designing your projects will help you appreciate UML as we will be covering UML after the Midterm.

Resources to utilize:

- Course Textbook
  - Chapter 2 has some gardening related examples
  - Chapters 3 and 4 for concepts
- Your requirements analysis

Implementation:

Implement your design via the Java programming language. We will be running your program and watch your growing plan work for a given amount of time. Your TAs and I will play the role of human gardeners interacting with your system's growing plan.

Some requirements:

- You must have a user interface. It does not need to be a graphical one but a graphical user interface will help with usability and thus make your TAs very happy! The happier your TAs are during grading the better your chances are for getting a higher grade!!! Poor UI designs will lose a lot of points!

- You must have several Modules. An implementation that does not include various standalone modules will not receive a good grade! Some module examples are: watering system, heating system, UI, log keeping, etc.
- Your garden must be a big garden with many sprinklers, heaters, sensors, plants of various kind, insects, etc.
- You must account for the passage of time. Meaning:
    - model every day as 24 minutes
    - sprinklers, heaters, etc. need to turn on and off as scheduled or as needed due to circumstances
    - plants grow and thus need more water, fertilizer, heat, etc. as the program progresses
- You must implement some randomness into your system. For example, it rains and so you should not turn on the sprinklers that day; or it gets really cold one night and you will need to turn on the heater even though it is during a warm time of the year; or maybe your garden is attacked by a pest and needs pesticides. Ensure at minimum 5 different types of random situations that could occur. The number of times or when each occurs should be random.
- You must implement manual overrides for a human gardener. Meaning the system must allow user interactions and interventions with the growing plan. The user should be able to update the plan as need be and the system should give the user appropriate warnings when decisions alter the plan.
- Your system must keep a detailed log of every occurrence, interaction, etc. as well as the progress of your growing plan and state of the garden/plants. We will run your program for a whole day (roughly equivalent to a whole month for your garden) and then read through the logs. If the logs are poor in quality and hard to navigate then even though your program may be working correctly, since we won't be able to determine that, we will assume that it does not work properly and thus you will lose significant amounts of points. You will not get a chance to remedy this by explaining your log or program.

Resources:
- Course "Java Programming" textbook
- Java: A Beginner's Tutorial (ISBN-13: 978-0992133047)
- For UI
    - Chapter 11 of the Java Programming Textbook
    - Java: A Beginner's Tutorial (ISBN-13: 978-0992133047) chapters 22 – 26
    - The Design of Everyday Things (ISBN-13: 978-0-465-06710-7)
    - Java FX:
        - Introducing Java FX 8 Programming (ISBN-13: 978-0071842556)
        - Java: A Beginner's Tutorial (ISBN-13: 978-0992133047) chapters 25 and 26
        - JavaFX For Dummies (ISBN-13: 978-1118385340)

Documentation:

Please provide ample documentation for your implementation. If the TAs and I can't figure out how to use your system then you will definitely lose a lot of points! You will not get the chance to showcase or verbally explain your code. Your only chance is to make the user interface as usable and simple as possible and to provide a robust help/user manual along with your implementation. We will refer to it during grading.

Also, ensure your log is highly readable. Provide a help guide/manual for your log if need be. If your logging system fails, is incoherent, or hard to navigate you will lose a substantial amount of points. You will not get the chance to showcase or verbally explain your log file either.

Notes:

- Cutting corners in any of the steps will result in massive point loss!
- Grading will be highly subjective! If your TAs or I find it difficult to use your software / read your log file or if we find either to be of very low quality, you will most definitely not receive a passing grade on the project.
- Reference any material you use to create your system. Any detected plagiarization will result in an instant F grade on the project and perhaps even further disciplinary actions by the department! We take this most seriously!!!