You are responsible for material covered in class sessions and individual assignments; but, here's a quick overview of especially important material related to this upcoming quiz.

You are permitted to bring into the quiz a single piece of paper (8.5" by 11") on which you have handwritten whatever you wish on one or both sides. This paper must include your name, it must be handwritten by you, and it will not be returned.

– Other than this piece of paper, the quiz is closed-note, closed-book, and closed-computer.

This is a quiz on `lejos.robotics.navigation`, `lejos.robotics.objectdetection`, and `lejos.robotics.subsumption`, including event handling, `DifferentialPilot`, and behavior-based robotics.

– You are expected to still follow Java conventions and class coding standards in your quiz answers.

### Interfaces and types

- Make sure you understand: for Java classes, a variable declared to be an instance of that class is considered to have that class as one of its types --
  - …BUT it is also considered to have as additional types that class's superclass and any ancestors,
  - …AND it is also considered to have as additional types any interfaces that the class implements

- Why is this included with these other topics for this quiz? Because you often end up implementing a Java interface in setting up an event listener!
  - …and an instance of such a class, implementing an event listener interface, can then be used as the argument for an add-listener method

- A class `MyClass` is to implement a given interface `DesiredInterface`.
  - Be able to write the class header for class `MyClass`.
  - At least what methods must now be implemented within the class `MyClass`?

- Remember that if a Java class is not explicitly declared to be a subclass of another class (using `extends`), then it is automatically a subclass of the overall Java superclass `Object`

- Does Java permit multiple inheritance?

### Event handlers and event listeners

- What is an event? What is an event source? What is an event listener?

- How can you add a listener to a `Button` instance?
  - What method can you use to register a listener of type `ButtonListener` for a `Button` instance?

- Be able to write a class implementing the `ButtonListener` interface.
  - Be able to create an instance of such as class
What methods must be implemented by a class implementing the `ButtonListener` interface?

Be able to add an instance -- anonymous or named -- of such as class to serve as an event listener for a given button

- When will the methods of a variable of type `ButtonListener` be called?
- Example of note - Week 4 Lab's `ButtonEventPlay.java` (tried out and extended in Project 2)
- What is a private inner class? What is an anonymous inner class?

**DifferentialPilot**

- Remember that different types of robots can be controlled by higher-level classes -- for example, a 2-wheeled robot, where the 2 wheels are each controlled by an independent motor, can be controlled using package `lejos.robotics.navigation`'s `DifferentialPilot` class.
- What are some of the methods of package `lejos.robotics.navigation`'s `DifferentialPilot` class?
- What are the arguments to the `DifferentialPilot` constructor?
  - how do the units used in this constructor's wheel diameter and track width arguments affect the arguments for methods such as `travel`?
  - what is meant by "track width"?
- Examples of note: Week 4 Lab's `TravelTest.java`, Week 5 Lecture 1's `SquareTracer.java` (tried out and extended in Project 2)

**RangeFeatureDetector**

- What are the arguments to the `RangeFeatureDetector` constructor? What is the meaning of each?
- Example of note: Week 5 Lecture 1's `ObjectDetectPlay.java` (tried out and extended in Project 2)
- What are some of the methods of package `lejos.robotics.objectdetection`'s `RangeFeatureDetector` class?
- If you create a `RangeFeatureDetector` instance based on an ultrasonic sensor, and add a `FeatureListener` instance to it so that the ultrasonic sensor is indeed sensitive to feature detection events, what method will be called when a feature is indeed detected?
- Assume you have a statement such as:
  ```java
  Feature result = myFeatureDetector.scan();
  ```
  - What will `scan` return if NO feature was detected?
  - What will `scan` return if a feature WAS detected? What is an example of a method that you could call on that returned object?

**LightSensor**

- Note that package `lejos.nxt`'s `LightSensor` class includes the methods:
  - `public void calibrateLow()` - call this method when the light sensor is reading the low value --
used by `readValue`

- `public void calibrateHigh()` - call this method when the light sensor is reading the high value --
  used by `readValue`

- `public int readValue()` - get the light reading

- **Note** that if `calibrateLow` and `calibrateHigh` have been called on a `LightSensor` instance, subsequent `readValue` calls on that `LightSensor` will be affected --
  - ..."dark" values read will return a value in the vicinity of 0
  - ..."light" values read will return a value in the vicinity of 100

- (where "dark" is based on what the light sensor read during its `calibrateLow` call, and "light" is based on what the light sensor read during its `calibrateHigh` call)

**Example** of note: Project 3 - Stage 3's `LightSensorTest2.java`

---

**Behavior-based robotics - lejos.robotics.subsumption**

- Make sure you understand the basic ideas of behavior-based robotics.

- In this approach, what is a behavior? What is the arbitrator? What does the arbitrator do?

- What are the key parts of the package `lejos.robotics.subsumption`?

- If a class extends the `Behavior` interface, what methods must it implement?

- How do you set up an `Arbitrator` instance?
  - What argument does its constructor expect?
  - What does an `Arbitrator` instance then do with that argument?
  - What is the significance of the order of the contents of the `Arbitrator` constructor's argument?

- You should be able to declare an array of `Behavior` instances.
  - For any Java array: what is the index of its first element?
  - For any Java array: what data field can you use to get that array's length (its number of elements)?
  - For any Java array: how can you write an expression whose value is a particular element within that array?

- What does an `Arbitrator` instance assume is the highest-priority `Behavior`?

- In subsumption architecture/behavior-based robotics as implemented in the `lejos.robotics.subsumption` package, when should an arbitrator call a behavior's `action` method?

- Consider a robot with a downward-facing light sensor whose goal is to follow the left edge of a thick black line on a white background.
  - what behavior might be appropriate if the light sensor reads too light?
  - what behavior might be appropriate if the light sensor reads too dark?
  - what behavior might be appropriate if the light sensor reads neither too light nor too dark?
  - make sure this is clear: in this approach, the idea is that these should be three separate, independent
classes implementing the `Behavior` interface (not one or two more complex classes -- the idea is to have more interacting simple behaviors rather than fewer more-complex classes)

- which `Behavior` interface method would an `Arbitrator` call to determine if that behavior should be considered?
  
  ...So, in the case of one of these behaviors, what method should you be calling in this method?

- If an `Arbitrator` determines that a `Behavior` should take place, what method of that `Behavior` will it call?

  ...So, in the case of one of these behaviors, what are some examples of methods that you might consider calling in this method?