Milestone 2: Make the spaceship move in a straight line and turn.

Note that you do not need to add in zero-gravity effects. That is, it's fine if your spaceship just moves forward at a constant speed when the key for that is pressed. It should also turn when the appropriate keys are pressed.

Decide which keys the user should press to have the spaceship move forward, and turn left and right. If you user keys other than the arrow keys, put this in the comment when you submit your code, so that I know how to test it.

Create a method `public void move()` in your `Ship` class that will change the position of your ship if the forward key is being held, and change the rotation of your ship if either of the turn keys are being held. Call this method in `Asteroids.paint()`.

What variable holds your ship's current position? How do you access the x coordinate and y coordinate of its current position? Note that all the relevant instance variables are public. This means you can call them in other classes without getters or setters, but you can also make them private and add getters and setters. It's up to you.

Add boolean instance variables for the forward, left and right turn keys to your `Ship` class that can be set to true if the appropriate key is being held down, and to false otherwise. For now, set the forward key variable to true, and the other two to false.

Add code to your `move()` method to make your `Ship` move forward when the forward key variable is true.

Once you have your ship moving, make sure that if it moves off one side of the screen, the appropriate position variable wraps around so that the ship appears on the other side of the screen.

At this point, your ship might not be moving in the direction that it is pointing, but we will fix this after we add in the `KeyListener` interface.

As discussed in class, we will make `Ship` implement the `KeyListener` interface (https://docs.oracle.com/javase/7/docs/api/java/awt/event/KeyListener.html) so that it can respond to keys on the keyboard being pressed. To do this, we need to change the header of the `Ship` class appropriately, and add three methods: `public void KeyPressed(...)`, `public void KeyReleased(...)`, and `public void KeyTyped(...)`. Even though we will leave `KeyTyped()` empty, it must be added to satisfy the conditions of the `KeyListener` interface.

All of the KeyListener methods take a KeyEvent object (https://docs.oracle.com/javase/7/docs/api/java/awt/event/KeyEvent.html) as their parameter. This object contains information about

which key was pressed or released. You can get the key code, which is a number representing the key pressed or released by calling the non-static method getKeyCode() for the KeyEvent instance that was passed in. The list of the constants corresponding to the key codes is on the KeyEvent API page (https://docs.oracle.com/javase/7/docs/api/java/awt/event/KeyEvent.html). Fill in the `KeyPressed()` and `KeyReleased()` methods so that when your chosen keys are pressed or released, their corresponding boolean variables change appropriately. See the class Test, which we did in class, for an example.

The `Canvas` object, which is the superclass of `Game` and hence `Asteroids`, generates `KeyEvents` when a key is pressed, released, or typed. We hence need to register our new `KeyListener` with `Asteroids`, so that it will know where to send these event objects. We do this by adding the code `this.addKeyListener(whateverYourShipInstanceIsCalled);` in the constructor of `Asteroids`.

Set the default for your boolean variable for the forward key being pressed to false. Test your code to make sure that your ship moves when the forward key is pressed (again it might still be moving in the wrong direction).

Add code in `move()` to make the `Ship` rotate if the appropriate key is being held. Test that the ship is rotating (but it may still move in the same wrong direction).

Now we will get your ship moving in the direction it is facing:

If your ship doesn't move in the correct direction even before pressing a turn key, then you need to change the direction that your ship is facing when it starts.

If your ship moves in the correct direction when you start, but then keeps moving in that same direction even after you press a turn key, then you need to use some trigonometry make sure you are incrementing the x and y coordinates the correct amount to go in the desired direction. Specifically, if you are currently incrementing the x and y coordinates by the same amount, you need to instead multiple the increase in the x coordinate by `Math.cos(Math.toRadians(rotation))` and multiple the increase in the y coordinate by `Math.sin(Math.toRadians(rotation))`.