Arcade_sim - Adding Driving Controls (Last week's project)

1) Create the drivetrain.py file following the instructions in the readme.md file

2) Update robot.py with joystick

-- Update imports
-- Add code to instantiate the controller
-- Add code to instantiate the drivetrain
-- Add code to define default command for drivetrain

```
import drivetrain

# Instantiate any subystems
self._drivetrain = drivetrain.DriveTrain(self.led)

# Setup the default commands for subsystems
self._drivetrain.setDefaultCommand(
# A split-stick arcade command, with forward/backward controlled by the left
# hand, and turning controlled by the right.
    RunCommand(
        lambda: self._drivetrain.driveManually(
            self._driver_controller.getRawAxis(1),
            self._driver_controller.getRawAxis(0),
        ),
        [self._drivetrain],
    )
)
```

<< I think the axis values are reversed. May need to swap the [0] and [1] >>

Update the basic driving project with an LED Subsystem.

Requirements:

1. The LED should be a rainbow pattern when the robot is stopped
2. The LED should be green when the robot is driving forward
3. The LED should be red when the robot is driving backwards

The black bolded text is the code to be added.

---

1) Copy four files from the project LED_subsystem project (Pull from Github if needed)
- ledsubsystem.py
- constants.py
- defaultCommand.py
- setLEDgreen.py

Make all of the filenames lower case (ledsubsystem.py)

---

2) Update **robot.py**
- Import the ledsubsystem

import ledsubsystem

- Instantiate the ledsubsystem right before instantiating the drivetrain
- Update the drivetrain instantiate statement to reference the LED subsystem

 # Instantiate any subystems
self.led = ledsubsystem.LEDSubsystem()
self._drivetrain = drivetrain.DriveTrain(self.led)

---

3) Start the simulation

4) Enable the display of the LED subsystem by selecting "Hardware" at the "Glass" top menu and enabling the "Addressable LEDs"

Verify no errors are present. The LEDs will not change at this time.

---

5) Update **robot.py**

- Add import statement for the default command and setledgreen

from defaultcommand import DefaultCommand
import setledgreen

- Add a default command for the LEDsubsystem within robot.py after the code the creates the default command for the drivetrain.

self.led.setDefaultCommand(defaultcommand(self.led))

---

6) Update **drivetrain.py** to be able to control the LED subsystem

- Add in import statement for the LED subsystem

```
from ledsubsystem import LEDSubsystem
```

- Update the "__init__" statement to reference the LED subsystem

```
def __init__(self, led: LEDSubsystem) -> None:
    super().__init__()
    self.led = led
```

---

7) Start the simulation and enable "teleOperated" mode
8) Verify the simulated LEDs are displaying a rainbow

---

9) Update the ledsubsystem (**ledsubsystem.py**) to be able to display a solid red color by creating new functions "setColorRed" and "displayRed" by making functions similar to the "setColorGreen" and "displayRed". The value of hue for red is "0"

<< [Code not included] Copy the other functions and update them to display red >>

---

10) Update the **drivetrain.py** file to control the LED subsystem based on the "Forward" driving command. If forward speed is greater than (>) 0, display green, else if forward speed is less than (<) 0, display red, otherwise display rainbow.

```
if forward > 0:            # Going forward
    self.led.displayGreen()

elif forward < 0:          # Going backwards
    self.led.displayRed()

else:                      # stopped
   self.led.displayRainbow()
```

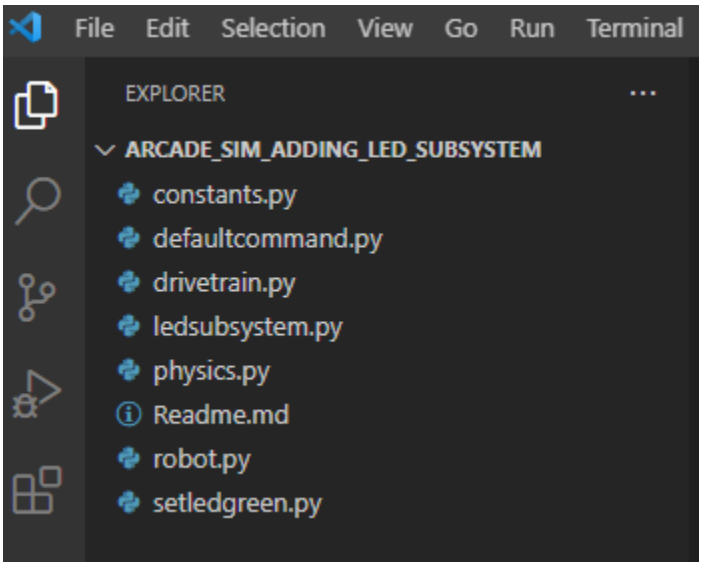
---

11) Within robot.py, update the drivetrain default command by reversing the X and Y axis. Swap the [0] and [1] to [1] and [0]

---

12) Start the simulation and enable "teleOperated" mode
13) Verify the simulated LEDs are displaying a rainbow
14) Press the "W" key and "S" key and verify the LEDs change colors as required

==========


File
Edit
Selection
View
Go
Run
Terminal
EXPLORER
ARCADE_SIM_ADDING_LED_SUBSYSTEM
constants.py
defaultcommand.py
drivetrain.py
ledsubsystem.py
physics.py
Readme.md
robot.py
setledgreen.py