Day 10 CS570

Test on Git/GitHub/OOP/TimedRobot

Name: _____

HC: _____

1. (6 points) Your robot is working well, but it doesn't have a climber. The code is checked and working, and is all consolidated on the `main` branch of the git repository. How would you use git to add new code to your repository?

2. (14 points) Write a class `Climber` class in python with following requirements (you don't have to worry about imports):

   - The Climber class initializers takes in two arguments: the first is the cancoder id of the motor that runs the climbing action. The second is an id number for the pneumatic actuator (solenoid) that releases the climber.
   - Instantiate a `Motor` class using as an input the cancoder id that was given in the initialization.
   - Instantiate a `Solenoid` using as an input the id that was given in the initialization. Then make sure the Solenoid is engaged by using the `set` method of the Solenoid and passing in the value `True`.
   - The class has a `release` method that releases the climber.
   - The class has a `climb` method that uses the `Motor`'s `run` method passing in the value 1.
   - Make it possible to print the `Climber` class to the console and report if it is deployed
     and the speed that the motor is set to by using the Motor's `get_speed` method.

3. (10 points) The `TimedRobot` class of `wpilib` has a method called `teleopPeriodic`. Describe what elements of the robot's code should be in this method.

4. (14 points) You are writing code for a robot, and have made a class for an `Arm` that your robot has. The code is in a file called `arm.py`. The arm can `tilt` and like the arm on the 7407 robot `extend`. The `tilt` method takes a number of degrees from 0 to 90 and tilts the arm from vertical that amount. The `extend` method takes a number from 0 to 1 and extends the arm based on the number. Full extension is 1 and completely pulled in is 0. You want to use this class to make the arm be tilted 45 degrees at the start of teleop. And you want to make sure the arm is fully extended at the start of autonomous. Add code below (including any necessary imports) to achieve those goals.

```
import os




import wpilib
from wpilib import TimedRobot

class MyRobot(TimedRobot):
    """
    Our default robot class, pass it to wpilib.run
    """


    def robotInit(self) -> None:




    def autonomousInit(self) -> None:




    def teleopInit(self) -> None:



    def teleopPeriodic(self) -> None:




if __name__ == "__main__":
    # If your ROMI isn't at the default address, set that here
```

```
os.environ["HALSIMWS_HOST"] = "10.0.0.2"
os.environ["HALSIMWS_PORT"] = "3300"

wpilib.run(MyRobot)
```

5. (12 points) Fill in the blanks.

```
import os

import _______________
from ____________ import TimedRobot, ____________, ____________
from wpilib.drive import ____________
from autonomous.controller import AutoControl


class Zee_Robot(TimedRobot):

        def robotInit(self):
            '''This method is called as the robot turns on and is often used to
            set up the joysticks and other presets.'''
            self.controller=_______________(0)
            self.left_motor=Spark(0)
            self.right_motor=Spark(1)
            self.drivetrain=DifferentialDrive(_________________, ___________________)
            self.autocontrol=AutoControl(self.drivetrain)

        def robotPeriodic(self):
            '''This is called every cycle of the code. In general the code is loop
                     through every .02 seconds.'''

            pass

        def autonomousInit(self):
            '''This is called once when the robot enters autonomous mode.'''
            pass

        def autonomousPeriodic(self):
            '''This is called every cycle while the robot is in autonomous.'''
            _______.______________.run()

        def teleopInit(self):
            '''This is called once at the start of Teleop.'''
            pass

        def teleopPeriodic(self):
            '''This is called once every cycle during Teleop'''
            forward=_________________.getRawAxis(0)
            rotate=___________________.getRawAxis(1)
            self.drivetrain.arcadeDrive(rotate, ____________________)
```

```
if __name__ == "__main__":
# If your ROMI isn't at the default address, set that here
    os.environ["HALSIMWS_HOST"] = "10.0.0.2"
    os.environ["HALSIMWS_PORT"] = "3300"
    wpilib.run(_______________________)
```

6. (6 points) Explain why it is important that all FRC robots are subclasses of some kind of robot that is described in `wpilib`. How do inheritance and polymorphism help make robot competitions work?

7. (12 points) Bob has written the code below. Please identify at least 3 mistakes in the code:

```
import wpilib from Spark, Solenoid


class Intake:

   def __init___(self):
      self.motor=Spark(motor_id)
      self.solenoid=Solenoid(1)


    def deploy(self, solenoid):
       self.solenoid.set(True)

    def retract(self, solenoid):
       solenoid.set(False)

    def run(self):
       motor.set_speed(1)
```