# Day 10 CS570

## Test Review on Git/GitHub/OOP/TimedRobot

1. Describe the purpose of a *branch* in git. What is the goal of a *main* branch?

**A branch in git is a way to work on a project without affecting the main code. The goal of the main branch is to have a stable version of the code that is ready to be used, while keeping track of the changes to the code in other branches.**

2. Write code for a claw object. The claw should have methods that allow it to open and close. The Claw object should be initializer should create a motor that the claw uses in its methods. Imagine that motor has a method called `run` that takes in a number between -1 and 1 to indicate how fast and in what direction to run the motor.

```
class Claw:

    def __init__(self, motor_id):
        self.motor=Motor(motor_id)

    def open(self):
        self.motor.run(1)

    def close(self):
        self.motor.run(-1)
```

3. The `TimedRobot` class of `wpilib` has a method called `robotInit`. Describe the what elements of the code should be in this method.

**The `robotInit` method is called when the robot is first turned on. It is often used to setup the joysticks and other presets that the robot will use. In the method you should create the joysticks and other objects that the robot will use to control the robot. These kind of objects might include subsystems like an elevator, shooter, intake, etc. It might also include objects that are more abstract like a class that tracks to position of the robot on the field and uses information from the robots drivetrain and vision systems.**

4. A friend of your writes code to run your robot. They have put that code in a file called `drivingwell.py`. Inside the file is a class called `DriveRobot`. The `DriveRobot` class has an `__init__` method that requires two numbers the first is the width of the robot in centimeters, and the second is the length of the robot in centimeters. Your robot is 55 cm wide and 60 cm long. What two lines of code are you going to add to your `robot.py` file to utilize your friends code after you have imported his file into the same directory as your `robot.py` file.

```
from drivingwell import DriveRobot

...

    self.driver=DriveRobot(55, 60)
```

5. Fill in the blanks.

```
import os

import ___wpilib______
from ___wpilib___ import TimedRobot, __Joystick__, __Spark__
from wpilib.drive import DifferentialDrive

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

class Tony_The_Robot(TimedRobot):
```

```
def robotInit(self):
    '''This method is called as the robot turns on and is often used to setup the joysticks and other p
    self.controller=__Joystick__(0)
    self.left_motor=Spark(0)
    self.right_motor=Spark(1)
    self.drivetrain=DifferentialDrive(_self.left_motor__, ___self.right_motor___)


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.'''
    pass

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=__self.controller___.getRawAxis(0)
    rotate=__self.controller___.getRawAxis(1)
    self.drivetrain.arcadeDrive(rotate, forward)
```

6. Ximena is making some changes to their robot code. They are currently on the `dev/shooter` branch of their git project that they are working on with other students. They have made significant changes, and have had an opportunity to test those changes on the robot and things seem to be working well. What should they do next to share their changes with other members of their team.

**Ximena should make a pull request to the main branch of the project. This will allow the other members of the team to review the changes that Ximena has made and to decide if they want to incorporate those changes into the main branch of the project.**

7. A team is considering several different types arms to score game pieces in their competition. The team decides to move forward with designing and coding two different arms. Describe how object oriented programming can support the development of two different kinds of arms. The programming lead wants to make it so that the code from either team can be utilized with the rest of the code. Describe how object oriented ideas of inheritance and polymorphism can be used to make this type of development possible.

**Object oriented programming can support the development of two different kinds of arms by creating a base class that has the common elements of the arms. This base class can be used to create two subclasses that have the unique elements of the two different arms. The programming lead can then use the base class to create objects of the two subclasses and use them in the code. This makes it so that the code from either team can be utilized with the rest of the code. Inheritance and polymorphism can be used to make this type of development possible by allowing the subclasses to have the same methods as the base class, but to have different implementations of those methods. This makes it so that the code can be written to use the base class, and then the objects of the subclasses can be used**

**in the code without changing the code.**

8. Write a class `Turret` class in python with following requirements.
    - The turret class initializers takes in two arguments the first is the cancoder of id of the motor that runs the turret, the second is a id number for the DIO port that the sensor is connected to.
    - Instantiate a `Motor` class using as an input of of the cancoder id that was given in the intialization.
    - Instantiate a `Sensor` using as an input the id that was given in the intialization.
    - The class has a `turn` method that takes in a number between -1 and 1 and uses the `Motor`'s `setspeed` method to set the speed of the turret.
    - The class has a `get_position` method that uses the `Sensor`'s `get_value` method to report the position of the turret.
    - Import the `Motor` and `Sensor` from `wpilib`
    - Make it possible to print the `Turret` class to the console and report the speed and position of the robot.

```python
from wpilib import Motor, Sensor

class Turret:

    def __init__(self, motor_id, sensor_id):
        self.motor=Motor(motor_id)
        self.sensor=Sensor(sensor_id)

    def turn(self, speed):
        self.motor.setspeed(speed)

    def get_position(self):
        return self.sensor.get_value()

    def __str__(self):
        return f'Turret: speed={self.motor.getspeed()}, position={self.get_position()}'
```

9. Describe the purpose of a *pull request* in git. What is the goal of a *pull request*?

**A pull request in git is a way to share changes that have been made in a branch with the main branch of the project. The goal of a pull request is to allow other members of the team to review the changes that have been made and to decide if they want to incorporate those changes into the main branch of the project.**

10. Discuss the benefits of creating subsystems in different files. How does this practice help with the development of a robot?

**Creating subsystems in different files helps to organize the code of the robot. It makes it so that the code is easier to read and understand. It also makes it so that the code is easier to maintain. If there is a problem with a subsystem, then the programmer can go to the file that has the subsystem and fix the problem without having to look through the entire code of the robot. This practice helps with the development of a robot by making it so that the code is easier to write and to debug. It also allows different members of the team to work on different files during development making merging of changes easier. It also makes it so that the code is easier to share with other members of the team.**

11. What is the point of the `super()` method in python?

**The `super()` method in python is used to call methods from the super class. This is useful when you want to use the methods from the super class in the subclass. For example, if you have a subclass that has a method that is the same as the method in the super class, you can use the `super()` method to call the method from the super class in the subclass. This is useful because it allows you to reuse code that is in the super class in the subclass.**

12. What is the purpose of the `__init__` method in a python class?

**The `__init__` method in a python class is called when an object of the class is created. It is used to initialize the object with the properties that it needs. For example, if you have a class that represents**

**a robot's odometry, you might want to initialize the robot with the position of the robot on the field. You would do this by setting the position of the robot in the odometry's `__init__` method.**

13. Find the three errors in the following code (There were actually four errors in the code).

```
import wpilib

class MyRobot(wpilib.TimedRobot):

    def robotInit(self):
        self.controller=wpilib.Joystick(0)
        self.left_motor=wpilib.Spark(**0**)
        self.right_motor=wpilib.Spark(1)
        **self**.drivetrain=wpilib.drive.DifferentialDrive(self.**left**_motor, self.right_motor)
        forward=self.controller.getRawAxis(0) # move this to the teleopPeriodic method

    def robotPeriodic(self):
        pass

    def autonomousInit(self):
        pass

    def autonomousPeriodic(self):
        pass

    def teleopInit(self):
        pass

    def teleopPeriodic(self):
        forward=self.controller.getRawAxis(0) # This came from the robotInit method

        rotate=self.controller.getRawAxis(1)
        self.drivetrain.arcadeDrive(rotate, forward)
```