Name: ____________________

Assessment
Runaway Rover

1) What does the "R" in the motor label "FS90R" stand for?
   a) Rapid
   b) Rotational
   c) Reverse
   d) Remote

2) What should you do first before attaching the motors to the wood?
   a) Paint them
   b) Test the wheels
   c) Calibrate the motors
   d) Charge the motors

3) Why do we need to calibrate the Servo motors before building the Rover?
   a) To change their color
   b) To make them smaller
   c) To connect them to the wheels
   d) To ensure they run at the correct speed and direction

4) When programming the Rover to go forward, why might it spin around instead?
   a) One motor might be disconnected.
   b) The code might need to be corrected.
   c) One motor might need to run in reverse to match the other.
   d) The wheels might be too small.

5) How can you make the Rover turn left?
   a) Make the right wheel go forward and the left wheel go backward
   b) Stop both motors
   c) Speed up both motors
   d) Disconnect one of the motors

6) Think about the steps you followed to build and program the Rover. How might these steps be similar to how engineers design and build real-life robots or vehicles? Give two examples.

P I P E R

Name: ____________________

7) Imagine you want to design a Rover to help clean up a park by picking up trash. Describe the features and functions you would include in your Rover to make it suitable for this task. How would you program it to perform its job effectively?

8) Look at the Python code below that makes the Rover move forward, turn left, move forward again, turn right, and then move forward one more time. How can you make the Rover stop for 2 seconds in the middle of its journey? Modify the code to make the Rover stop after it turns left for the first time and before it continues moving forward again.

Name: ____________________

Answer Key

Runaway Rover

1) B - Rotational

2) C - Calibrate the motors

3) D - To ensure they run at the correct speed and direction

4) C - One motor might need to run in reverse.

5) A - Make the right wheel go forward and the left wheel go backward

6) *Example:* Engineers also need to calibrate and test the motors and other components of real-life robots and vehicles to ensure they work correctly before assembly. Just like creating functions for different movements of the Rover, engineers program real-life robots and vehicles with specific instructions to perform various tasks, such as moving forward, turning, and avoiding obstacles.

7) *Example:*

- Features: The Rover would have an arm or claw to pick up trash, sensors to detect trash, and a compartment to store collected trash. It might also have wheels suitable for different terrains.
- Functions: The Rover would have functions to move forward, backward, turn left, and right. It would also have a function to operate the arm or claw to pick up trash.
- Programming: I would program the Rover to navigate the park in a systematic pattern, using sensors to detect and pick up trash. It would stop to collect trash when detected, store it in the compartment, and continue until the park is clean.

8) *Example:*

```
## ---- Imports ---- ##
import time
import board
from piper_blockly import *

## ---- Definitions ---- ##

GP0 = piperServoPin(board.GP0, "GP0")

try:
  set_digital_view(True)
except:
  pass
```

Name: ____________________

```
GP1 = piperServoPin(board.GP1, "GP1")
# Describe this function...
def go_forward():
  GP0.setServoFraction(min(max(30 * 0.0018 + 0.5,0),1))
  GP1.setServoFraction(min(max((-30) * 0.0018 + 0.5,0),1))

# Describe this function...
def go_left():
  GP0.setServoFraction(min(max((-30) * 0.0018 + 0.5,0),1))
  GP1.setServoFraction(min(max((-30) * 0.0018 + 0.5,0),1))

# Describe this function...
def go_backwards():
  GP0.setServoFraction(min(max((-30) * 0.0018 + 0.5,0),1))
  GP1.setServoFraction(min(max(30 * 0.0018 + 0.5,0),1))

# Describe this function...
def go_right():
  GP0.setServoFraction(min(max(30 * 0.0018 + 0.5,0),1))
  GP1.setServoFraction(min(max(30 * 0.0018 + 0.5,0),1))


## ---- Code ---- ##
go_forward()
time.sleep(1)
go_left()
time.sleep(0.5)

# Add a stop for 2 seconds
GP0.setServoFraction(0.5)  # Stop motor
GP1.setServoFraction(0.5)  # Stop motor
time.sleep(2)  # Pause for 2 seconds

go_forward()
time.sleep(1)
go_right()
time.sleep(0.5)
go_forward()
time.sleep(1)
```