**Lab 3: Clients/Servers**

1. **ROS2 Services: (25 points)**
   1. Let’s go back to the topic activity from Lab 2. Just to recap, the node “number\_publisher” publishes a number on the /number topic. The node “number\_counter” gets the number, adds it to a counter, and publishes the counter on the /number\_count topic. Here is what the original structure looked like. **Diagram

      Description automatically generated**
   2. You will add a functionality to reset the counter to zero:
      1. Create a service server inside the number\_counter node
      2. Service name: /reset\_counter
      3. Service type: example\_interfaces/srv/SetBool. Use ros2 interface show to learn what’s inside
      4. When the server is called, check the Boolean data from the request. If true, set the counter variable to 0
      5. Create your own custom node called “number\_resetter” to call this /reset\_counter service.
   3. Here is what the overall structure should look like:   
      Diagram

      Description automatically generated
2. **Fibonacci: (15 points)**
   1. The Fibonacci sequence, as a reminder, is a set of integers that starts with 0, 1, 1, and then a series of steadily increasing numbers. Each number is equal to the sum of the preceding two numbers. [0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, …]
   2. You may use the example service server for adding two integers together. You will write a new node called fibonacci.py that requests the add two ints service a total of 10 times. The first time, the two inputs will be 0 and 1. The second time, the input will be the sum from the first call plus 1. The third time, the input will be the sum from the second call plus the sum from the first call, etc. Before the node shuts down, output to the screen the whole Fibonnaci sequence for your 10 calls (the result should be the sequence listed in part a, up to 89).

An additional five points will be allocated in each homework assignment for style: Make sure your code is commented, neat, and variable names make sense. You should consult the Python style guide <https://peps.python.org/pep-0008/> as well as the ROS2 style guide: <https://docs.ros.org/en/rolling/The-ROS2-Project/Contributing/Code-Style-Language-Versions.html#python>.

**What to turn in**

* In a single zip file named DirectoryID\_Lab3.zip,
  + Your entire package that contains Q1 and Q2.
  + A video walkthrough showing you running Q1 and Q2.