Task 1.1 – Publish and Subscribe data on ROS

The objective of this task is to familiarize you with the framework of the Robot Operating System by implementing your own simple publisher and subscriber nodes in Python. You can **visit** [this link] for **better understanding of publishers and subscribers**.

Please find the ***node1.py, node2.py*** *and* ***node3.py*** files in the ***"scripts"*** folder. Add your code in these files to accomplish the following:

Given:

We hope that you have already learned the basic concepts of ROS. You should be able to understand the difference between **node** and **topic** in ROS. Refer to the graph in Figure 1.

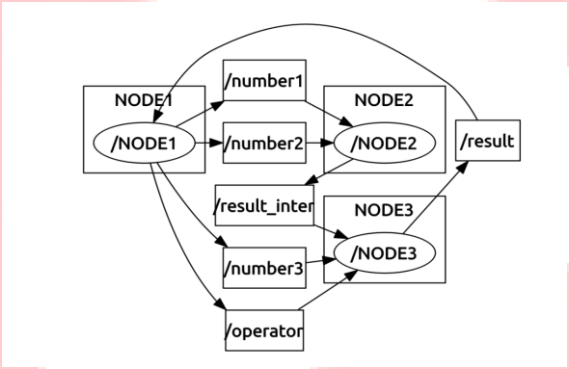

Figure 1: Graph representation

NOTE: Arrow leaving a node denotes a publisher and arrow entering a node denotes a subscriber.

There are three nodes NODE1, NODE2, NODE3 as shown in the graph. These are described below:

1. **NODE1**: This node **publishes four topics** namely "/number1", "/number2", "/number3" and "/operator" and **subscribes one topic** namely "/result". The topic "/result" is published by **NODE3**. **The three numbers which are published over the topics "/number1", "/number2" and "/number3" can be chosen in random or can follow a particular order and the same rule is applicable in choosing the operator.**
2. **NODE2**: This node **has two subscribers and one publisher**. **NODE2 subscribes** two topics "/number1" and "/number2" which are published by **NODE1**. After applying addition operation, it publishes that number on topic "/result_inter".
3. **NODE3:** This node has **three subscribers and one publisher**.
    - "/number3" and "/operator": These topics are published by **NODE1** and subscribed by **NODE3**.
    - "/result_inter": This topic is published by **NODE2** and subscribed by **NODE3**.
    - "/result": This topic is published by **NODE3** and subscribed by **NODE1**.

Topics' data-types are as follows:

1. **"/number1", "/number2" , "/number3" and "/result_inter"**: Integer; value can be any number
2. **"/operator"**: String; value of operator can **only** be "**add**", "**mul**", "**sub**" or "**div**"
3. **"/result"**: float; value of this topic can be any float value

Problem Statement:

As given in the graph, there are three nodes which perform different tasks at a particular point in time.

1. **NODE1** publishes three numbers and an operator. Two of these numbers have to be subscribed by **NODE2**. Teams have to modify the ***node1.py*** file in the ***"scripts"*** folder to publish four topics and subscribe the topic "/result".
2. **NODE2** subscribes two topics published by **NODE1** and an **addition** operation has to be done on them (data of "/number1" and "/number2"). The result after an operation on numbers has to be published on topic "/result_inter". Teams have to modify the ***node2.py*** file in ***"scripts"*** folder to take data from topics "/number1" and "/number2".After applying the addition operation result should be published to topic "/result_inter".
3. **NODE3** subscribes three topics, two of them are operands and one is the operator. One operand is published by **NODE1** on topic "/number3"and the second operand is published by **NODE2** on topic "/result_inter". Based on the operator published by **NODE1**, the specific operation has to be done between the two operands and the final result has to be sent back to **NODE1**. The operation list is given as "**add**" for addition, "**sub**" for subtraction, "**mul**" for multiplication, and "**div**" for division. Teams have to modify the ***node3.py*** file in "***scripts"*** folder to take data from NODE1 and NODE2 as specified above and return the result back to NODE1.

You are supposed to write a launch file to run all these nodes together. Create a launch file in your package with name **task1.launch** and add the following contents in it.

```
<launch>

<node name="NODE1" pkg="sample_package" type="node1.py"
output="screen"/>
<node name="NODE2" pkg="sample_package" type="node2.py" />
<node name="NODE3" pkg="sample_package" type="node3.py" />

</launch>
```

Expected output of task_1.launch file in the terminal:

The output of **NODE1** has to be of the form:

(number1 + number 2) operator number3 = result

NOTE: Launch file prints the value of NODE1 only. Make necessary changes in NODE1(node1.py) for the desired output.

```
[INFO] [WallTime: 1477475735.238887] (148+149) add 150 = 447.0
[INFO] [WallTime: 1477475736.238809] (149+150) sub 151 = 148.0
[INFO] [WallTime: 1477475737.239353] (150+151) mul 152 = 45752.0
[INFO] [WallTime: 1477475738.238655] (151+152) div 153 = 1.98039209843
[INFO] [WallTime: 1477475739.239473] (152+153) add 154 = 459.0
[INFO] [WallTime: 1477475740.239483] (153+154) sub 155 = 152.0
[INFO] [WallTime: 1477475741.238769] (154+155) mul 156 = 48204.0
[INFO] [WallTime: 1477475760.339008] (173+174) sub 175 = 172.0
[INFO] [WallTime: 1477475761.339459] (174+175) mul 176 = 61424.0
[INFO] [WallTime: 1477475762.339079] (175+176) div 177 = 1.98305082321
[INFO] [WallTime: 1477475763.339084] (176+177) add 178 = 531.0
[INFO] [WallTime: 1477475764.339821] (177+178) sub 179 = 176.0
[INFO] [WallTime: 1477475765.338862] (178+179) mul 180 = 64260.0
[INFO] [WallTime: 1477475766.338961] (179+180) div 181 = 1.9834253788
[INFO] [WallTime: 1477475767.339559] (180+181) add 182 = 543.0
[INFO] [WallTime: 1477475768.339105] (181+182) sub 183 = 180.0
[INFO] [WallTime: 1477475769.338907] (182+183) mul 184 = 67160.0
[INFO] [WallTime: 1477475770.339755] (183+184) div 185 = 1.98378384113
[INFO] [WallTime: 1477475771.338883] (184+185) add 186 = 555.0
[INFO] [WallTime: 1477475772.338886] (185+186) sub 187 = 184.0
[INFO] [WallTime: 1477475773.338236] (186+187) mul 188 = 70124.0
[INFO] [WallTime: 1477475774.338985] (187+188) div 189 = 1.98412692547
```

Figure 2: Expected Output

You can run **rqt_graph** command in terminal to check your **nodes** and **topics relations**. The output of graph should be **similar to Figure 1**.

After completing the given task store your whole package into Task-1 folder with name <team_id>task1.1. Instructions for uploading your task will be provided on the portal.