Embedded Visual Control

Assignment 1

Grade: Individual (10%)	Deadline: Monday 28/04 23:59
Title: Introduction to ROS	

Instructions

After launching the template ROS node, you can develop your own nodes. Create a copy of the template_ws directory and name it worshop1_<student_id> under the workshops folder. Inside this dedicated workspace for the first Workshop, you will create a new package. This package will contain two nodes, one publishing node and one subscribing node. The nodes functionalities will be the following:

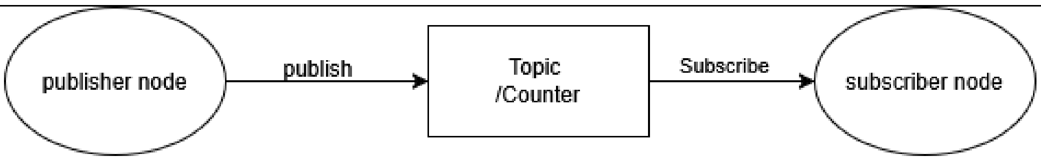
1. Publisher node: The publisher will send topic messages of type UInt8 using a publishing rate of 1Hz and a queue size of 1. The topic value will be an incrementing value from 0 to the maximum representable UInt8 value. When reaching the maximum representable number, the message should start again from 0.
2. Subscriber node: The subscriber will listen to the same topic with a queue size of 1.

Assessment Criteria:	**Submission Guidelines:**
● Short and concise answers ● Arguments must be supported by comprehensive visuals (and logging information) ● Code quality	● Each student must submit their personal workshop directory (in zipped format) with their report inside ● Answers are written in the space between questions

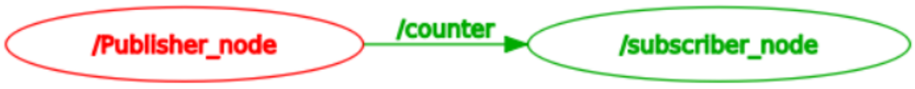
Name: Toan Verrijt Student number: 2222302

Questions

1. **Subscriber Functionality:** Make the callback of the subscriber trying to predict the newcoming value of the publisher knowing it will be always incrementing by 1.
    a. Illustrate the ROS architecture. (**1pts**)

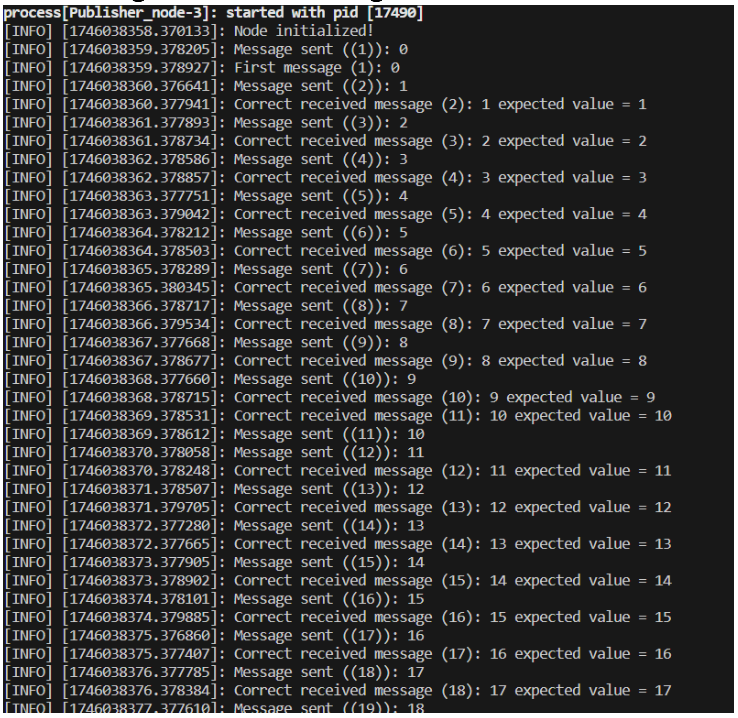


This was also tested and it seems to match the rqt_graph



    b. Log the expected and received values. For visualization purposes, log unexpected values as warnings. Plot the received message values w.r.t. time. (**1pts**)

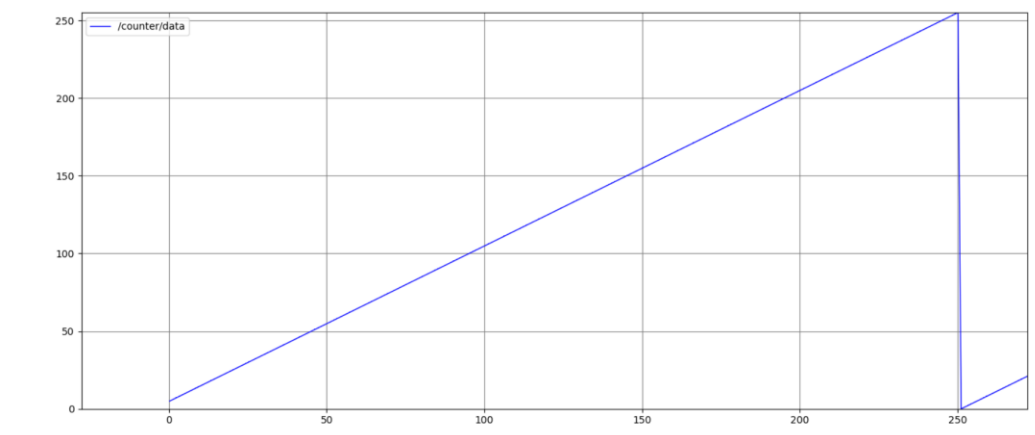
Terminal gives the following:



```
process[Publisher_node-3]: started with pid [17490]
[INFO] [1746038358.370133]: Node initialized!
[INFO] [1746038359.378205]: Message sent ((1)): 0
[INFO] [1746038359.378927]: First message (1): 0
[INFO] [1746038360.376641]: Message sent ((2)): 1
[INFO] [1746038360.377941]: Correct received message (2): 1 expected value = 1
[INFO] [1746038361.377893]: Message sent ((3)): 2
[INFO] [1746038361.378734]: Correct received message (3): 2 expected value = 2
[INFO] [1746038362.378586]: Message sent ((4)): 3
[INFO] [1746038362.378857]: Correct received message (4): 3 expected value = 3
[INFO] [1746038363.377751]: Message sent ((5)): 4
[INFO] [1746038363.379042]: Correct received message (5): 4 expected value = 4
[INFO] [1746038364.378212]: Message sent ((6)): 5
[INFO] [1746038364.378503]: Correct received message (6): 5 expected value = 5
[INFO] [1746038365.378289]: Message sent ((7)): 6
[INFO] [1746038365.380345]: Correct received message (7): 6 expected value = 6
[INFO] [1746038366.378717]: Message sent ((8)): 7
[INFO] [1746038366.379534]: Correct received message (8): 7 expected value = 7
[INFO] [1746038367.377668]: Message sent ((9)): 8
[INFO] [1746038367.378677]: Correct received message (9): 8 expected value = 8
[INFO] [1746038368.377660]: Message sent ((10)): 9
[INFO] [1746038368.378715]: Correct received message (10): 9 expected value = 9
[INFO] [1746038369.378531]: Correct received message (11): 10 expected value = 10
[INFO] [1746038369.378612]: Message sent ((11)): 10
[INFO] [1746038370.378058]: Message sent ((12)): 11
[INFO] [1746038370.378248]: Correct received message (12): 11 expected value = 11
[INFO] [1746038371.378507]: Message sent ((13)): 12
[INFO] [1746038371.379705]: Correct received message (13): 12 expected value = 12
[INFO] [1746038372.377280]: Message sent ((14)): 13
[INFO] [1746038372.377665]: Correct received message (14): 13 expected value = 13
[INFO] [1746038373.377905]: Message sent ((15)): 14
[INFO] [1746038373.378902]: Correct received message (15): 14 expected value = 14
[INFO] [1746038374.378101]: Message sent ((16)): 15
[INFO] [1746038374.379885]: Correct received message (16): 15 expected value = 15
[INFO] [1746038375.376860]: Message sent ((17)): 16
[INFO] [1746038375.377407]: Correct received message (17): 16 expected value = 16
[INFO] [1746038376.377785]: Message sent ((18)): 17
[INFO] [1746038376.378384]: Correct received message (18): 17 expected value = 17
[INFO] [1746038377.377610]: Message sent ((19)): 18
```

Sometimes the log is printed after a value is sent this looks weird in the terminal but the sequence is still correct. This means that the publisher can sent a messages, the subscriber can receive it print a log and after the publisher prints a log. This way it seems as if the value is received before it has been sent.

Rqt_plot shows the following :



c. Are they what you expect? Calculate the drop rate based on the expected messages and the received ones. (**1pts**)

The messages are as expected, since the subscriber can keep up with the publisher, the values should match and the drop rate should be 0.
The following formula is used to calculate the drop rate

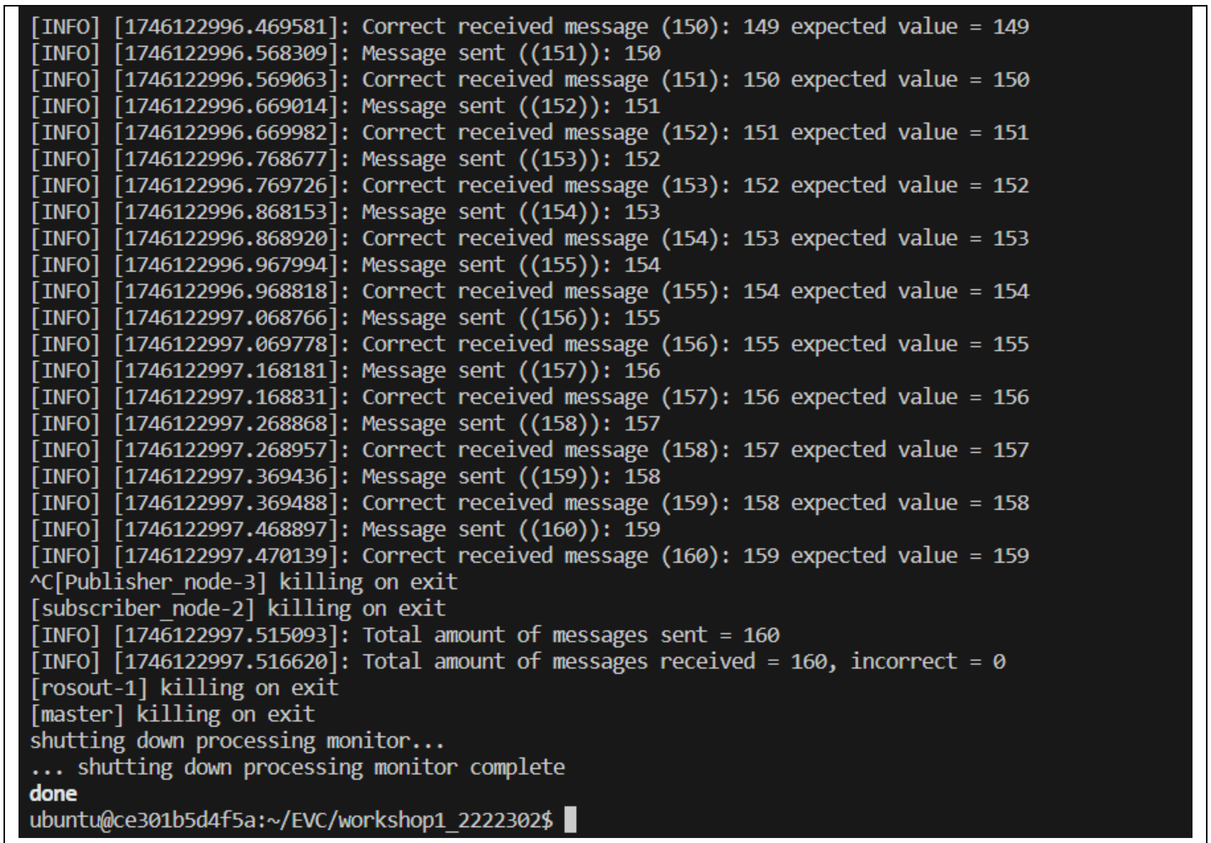
$$Drop\ rate\ [\%] = \left(\frac{incorrect\ message + missed\ messages}{Total\ messages}\right) * 100$$

Total messages : 457
incorrect messages : 0
missed : 0
All messages received in the tests where correct and 0 warnings where logged , making the drop rate 0.0% .

2. **Increasing Publish Rate:** Now increase the publish rate to 10Hz.

a. Does anything change related to the messages received? (**1pts**)

All the messages get correctly received and the correct values are shown since the publisher now sends messages 10 times as fast they are received 10 times as fast. So the only thing that changes is the speed



```
[INFO] [1746122996.469581]: Correct received message (150): 149 expected value = 149
[INFO] [1746122996.568309]: Message sent ((151)): 150
[INFO] [1746122996.569063]: Correct received message (151): 150 expected value = 150
[INFO] [1746122996.669014]: Message sent ((152)): 151
[INFO] [1746122996.669982]: Correct received message (152): 151 expected value = 151
[INFO] [1746122996.768677]: Message sent ((153)): 152
[INFO] [1746122996.769726]: Correct received message (153): 152 expected value = 152
[INFO] [1746122996.868153]: Message sent ((154)): 153
[INFO] [1746122996.868920]: Correct received message (154): 153 expected value = 153
[INFO] [1746122996.967994]: Message sent ((155)): 154
[INFO] [1746122996.968818]: Correct received message (155): 154 expected value = 154
[INFO] [1746122997.068766]: Message sent ((156)): 155
[INFO] [1746122997.069778]: Correct received message (156): 155 expected value = 155
[INFO] [1746122997.168181]: Message sent ((157)): 156
[INFO] [1746122997.168831]: Correct received message (157): 156 expected value = 156
[INFO] [1746122997.268868]: Message sent ((158)): 157
[INFO] [1746122997.268957]: Correct received message (158): 157 expected value = 157
[INFO] [1746122997.369436]: Message sent ((159)): 158
[INFO] [1746122997.369488]: Correct received message (159): 158 expected value = 158
[INFO] [1746122997.468897]: Message sent ((160)): 159
[INFO] [1746122997.470139]: Correct received message (160): 159 expected value = 159
^C[Publisher_node-3] killing on exit
[subscriber_node-2] killing on exit
[INFO] [1746122997.515093]: Total amount of messages sent = 160
[INFO] [1746122997.516620]: Total amount of messages received = 160, incorrect = 0
[rosout-1] killing on exit
[master] killing on exit
shutting down processing monitor...
... shutting down processing monitor complete
done
ubuntu@ce301b5d4f5a:~/EVC/workshop1_2222302$ 
```

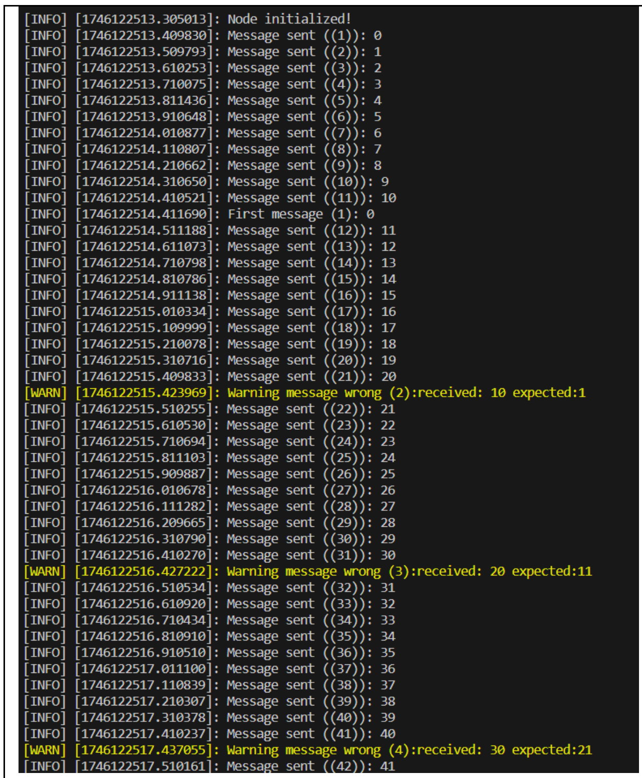
b. Calculate the drop rate. (**1pts**)

Since all the messages get correctly received the drop rate is 0.0%

In the test above this is also the case
Total messages : 160
incorrect messages : 0
missed : 0
$Drop\ rate\ [\%] = \left(\frac{0}{160}\right) * 100$ = 0.0%

Name: Toan Verrijt Student number: 2222302

3. **Increasing Subscriber Execution Time:** Add artificial delay to the subscriber's callback to achieve a service rate of 1Hz.
    a. Visualize the received message values. (1pts)

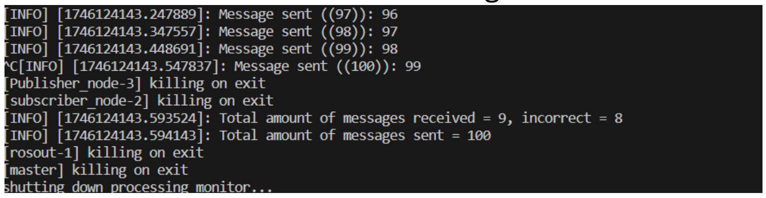


```
[INFO] [1746122513.305013]: Node initialized!
[INFO] [1746122513.409830]: Message sent ((1)): 0
[INFO] [1746122513.509793]: Message sent ((2)): 1
[INFO] [1746122513.610253]: Message sent ((3)): 2
[INFO] [1746122513.710075]: Message sent ((4)): 3
[INFO] [1746122513.811436]: Message sent ((5)): 4
[INFO] [1746122513.910648]: Message sent ((6)): 5
[INFO] [1746122514.010877]: Message sent ((7)): 6
[INFO] [1746122514.110807]: Message sent ((8)): 7
[INFO] [1746122514.210662]: Message sent ((9)): 8
[INFO] [1746122514.310650]: Message sent ((10)): 9
[INFO] [1746122514.410521]: Message sent ((11)): 10
[INFO] [1746122514.411690]: First message (1): 0
[INFO] [1746122514.511188]: Message sent ((12)): 11
[INFO] [1746122514.611073]: Message sent ((13)): 12
[INFO] [1746122514.710798]: Message sent ((14)): 13
[INFO] [1746122514.810786]: Message sent ((15)): 14
[INFO] [1746122514.911138]: Message sent ((16)): 15
[INFO] [1746122515.010334]: Message sent ((17)): 16
[INFO] [1746122515.109999]: Message sent ((18)): 17
[INFO] [1746122515.210078]: Message sent ((19)): 18
[INFO] [1746122515.310716]: Message sent ((20)): 19
[INFO] [1746122515.409833]: Message sent ((21)): 20
[WARN] [1746122515.423969]: Warning message wrong (2):received: 10 expected:1
[INFO] [1746122515.510255]: Message sent ((22)): 21
[INFO] [1746122515.610530]: Message sent ((23)): 22
[INFO] [1746122515.710694]: Message sent ((24)): 23
[INFO] [1746122515.811103]: Message sent ((25)): 24
[INFO] [1746122515.909887]: Message sent ((26)): 25
[INFO] [1746122516.010678]: Message sent ((27)): 26
[INFO] [1746122516.111282]: Message sent ((28)): 27
[INFO] [1746122516.209665]: Message sent ((29)): 28
[INFO] [1746122516.310790]: Message sent ((30)): 29
[INFO] [1746122516.410270]: Message sent ((31)): 30
[WARN] [1746122516.427222]: Warning message wrong (3):received: 20 expected:11
[INFO] [1746122516.510534]: Message sent ((32)): 31
[INFO] [1746122516.610920]: Message sent ((33)): 32
[INFO] [1746122516.710434]: Message sent ((34)): 33
[INFO] [1746122516.810910]: Message sent ((35)): 34
[INFO] [1746122516.910510]: Message sent ((36)): 35
[INFO] [1746122517.011100]: Message sent ((37)): 36
[INFO] [1746122517.110839]: Message sent ((38)): 37
[INFO] [1746122517.210307]: Message sent ((39)): 38
[INFO] [1746122517.310378]: Message sent ((40)): 39
[INFO] [1746122517.410237]: Message sent ((41)): 40
[WARN] [1746122517.437055]: Warning message wrong (4):received: 30 expected:21
[INFO] [1746122517.510161]: Message sent ((42)): 41
```

Every messages except the first one is received incorrectly.

    b. Calculate the drop rate. (**1pts**)

Only the fist messages is correct because the subscriber can't keep up with the publisher.
This means that the amount of incorrect or not received messages only increases while the number of correct will stay at one. This means the drop rate will go to 100%
If we use the following formula :
$\left[Drop\ rate\ [\%] = \left(1 - \frac{correct\ messages}{Total\ messages}\right) * 100 \right]$ and take the limit where the total amount of messages goes to infinity the answers is 100%
In the actual test where 100 messages where sent :



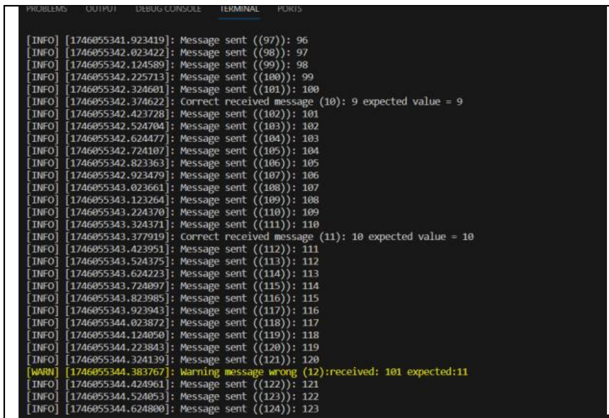
```
[INFO] [1746124143.247889]: Message sent ((97)): 96
[INFO] [1746124143.347557]: Message sent ((98)): 97
[INFO] [1746124143.448691]: Message sent ((99)): 98
^C[INFO] [1746124143.547837]: Message sent ((100)): 99
[Publisher_node-3] killing on exit
[subscriber_node-2] killing on exit
[INFO] [1746124143.593524]: Total amount of messages received = 9, incorrect = 8
[INFO] [1746124143.594143]: Total amount of messages sent = 100
[rosout-1] killing on exit
[master] killing on exit
shutting down processing monitor...
```

Correct = 1
Total = 100

$$Drop\ rate\ [\%] = \left(1 - \frac{1}{100}\right) * 100 = 99.0\%$$

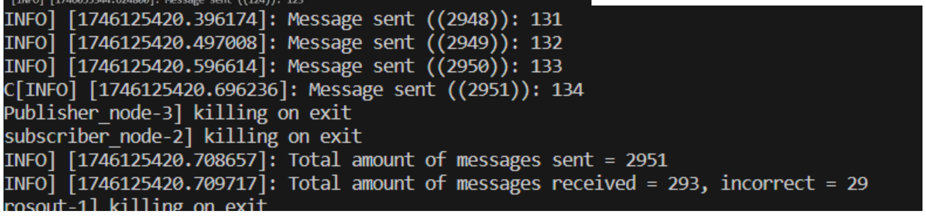
Name: Toan Verrijt Student number: 2222302

4. **Increasing Subscriber's queue:** Increase the size of the subscriber queue to 10 items.
   a. What happens to the logged messages? Calculate the drop rate. (**1pts**)



PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
[INFO] [1746055341.923419]: Message sent ((97)): 96
[INFO] [1746055342.023422]: Message sent ((98)): 97
[INFO] [1746055342.124589]: Message sent ((99)): 98
[INFO] [1746055342.225713]: Message sent ((100)): 99
[INFO] [1746055342.324601]: Message sent ((101)): 100
[INFO] [1746055342.374622]: Correct received message (10): 9 expected value = 9
[INFO] [1746055342.423728]: Message sent ((102)): 101
[INFO] [1746055342.524704]: Message sent ((103)): 102
[INFO] [1746055342.624477]: Message sent ((104)): 103
[INFO] [1746055342.724107]: Message sent ((105)): 104
[INFO] [1746055342.823363]: Message sent ((106)): 105
[INFO] [1746055342.923479]: Message sent ((107)): 106
[INFO] [1746055343.023661]: Message sent ((108)): 107
[INFO] [1746055343.123264]: Message sent ((109)): 108
[INFO] [1746055343.224370]: Message sent ((110)): 109
[INFO] [1746055343.324371]: Message sent ((111)): 110
[INFO] [1746055343.377919]: Correct received message (11): 10 expected value = 10
[INFO] [1746055343.423951]: Message sent ((112)): 111
[INFO] [1746055343.524375]: Message sent ((113)): 112
[INFO] [1746055343.624223]: Message sent ((114)): 113
[INFO] [1746055343.724097]: Message sent ((115)): 114
[INFO] [1746055343.823985]: Message sent ((116)): 115
[INFO] [1746055343.923943]: Message sent ((117)): 116
[INFO] [1746055344.023872]: Message sent ((118)): 117
[INFO] [1746055344.124050]: Message sent ((119)): 118
[INFO] [1746055344.223843]: Message sent ((120)): 119
[INFO] [1746055344.324139]: Message sent ((121)): 120
[WARN] [1746055344.383767]: Warning message wrong (12):received: 101 expected:11
[INFO] [1746055344.424961]: Message sent ((122)): 121
[INFO] [1746055344.524053]: Message sent ((123)): 122
[INFO] [1746055344.624800]: Message sent ((124)): 123
```



```
INFO] [1746125420.396174]: Message sent ((2948)): 131
INFO] [1746125420.497008]: Message sent ((2949)): 132
INFO] [1746125420.596614]: Message sent ((2950)): 133
C[INFO] [1746125420.696236]: Message sent ((2951)): 134
Publisher_node-3] killing on exit
subscriber_node-2] killing on exit
INFO] [1746125420.708657]: Total amount of messages sent = 2951
INFO] [1746125420.709717]: Total amount of messages received = 293, incorrect = 29
rosout-1] killing on exit
```

If we look at the messages received and how many are wrong this has a factor of 10 between them matching the seen behavior.

Total = 2951

Received = 293

Incorrect = 29

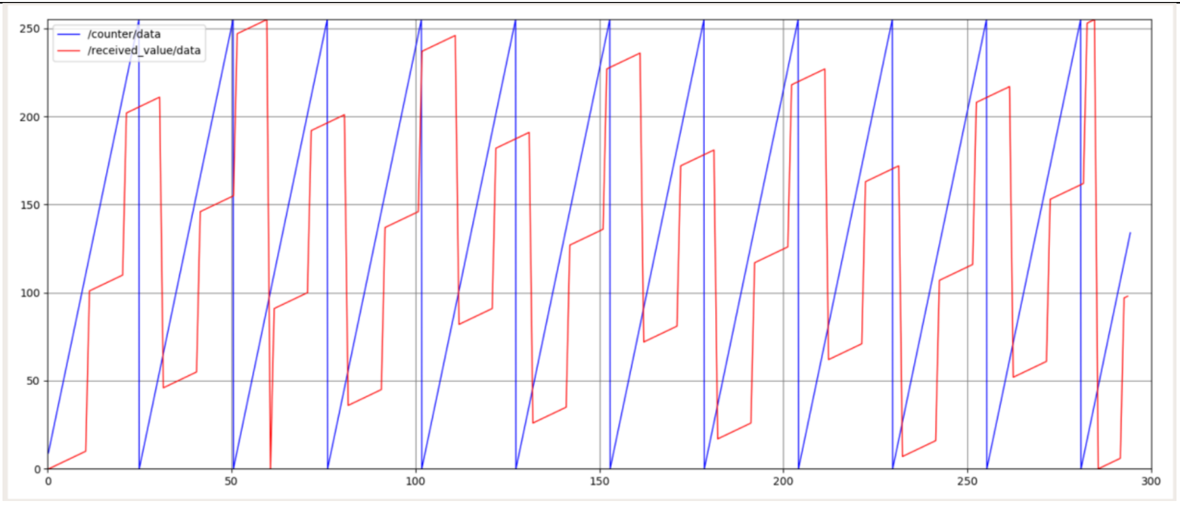
Missed = 2951 – 293 = 2658

$$Drop\ rate\ [\%] = \left(\frac{incorrect\ message + missed\ messages}{Total\ messages}\right) * 100$$

$$Drop\ rate\ [\%] = \left(\frac{29 + 2658}{2951}\right) * 100 = 91.1\%$$

   b. Visualize the received message values. Can you notice a pattern in the messages correctly predicted and dropped? (**2pts**)

Name: Toan Verrijt Student number: 2222302



Yes the pattern shows that the received values increment by one for the queue size of 10 and then spontaneously change value because of course a drastically different value was added. The queue gets filled with the next correct values and afterwards with a complete different one. This matches what was seen in the drop rate calculation. This means its stays correct for a bit, changes completely introducing an incorrect value whereafter it fixes it self for the upcoming 9 values.