Part 0:

What is a ros publisher ?

What is a ros subscriber ?

List 3 ros message types ?

Part 1: A simple publisher

Write a simple ros publisher in python that publish on the topic /counter a number that is increment each second. You can use rospy.sleep(1) to sleep for one second.

Part 2: Obstacle avoidance

In this exercice you have to code a simple obstacle avoidance system for the turtlebot. Here is a video of the expected result: https://youtu.be/nsdixTXz4V0

Step 1

Launch the turtle bot in the gazebo simulator using the following command:

```
TURTLEBOT_GAZEBO_WORLD_FILE=/opt/ros/kinetic/share/turtlebot_gazebo/worlds/corridor.world
roslaunch turtlebot_gazebo turtlebot_world.launch
```

Question 1:

What are the topics published by the turtlebot ?

Question 2:

Visualize the turtlebot and the sensors in rviz.

Question 3:

- What is the type of the messages on /scan ?
- Use the commande line tools to display a message on /scan.
- Visualize /scan topic in rviz

Question 4:

Move the robot by using the rostopic command. In an other terminal print the /scan topic ?

Question 5:

In order to avoid obstacles, you are going to use the messages available on the /scan topic. They correspond to the depth information projected on a 2D plan. Describe the step that you want to follow in order to achieve the obstacle avoidance behavior.

Question 6:

Complete the following code to implement a similar behavior to the one displayed in the video.

Tips: - To get the minimum value of a list you can use min(my_list) in python. (warning this function will fail if the list is empty. Use a try except block to handle this case).

```
import rospy

from geometry_msgs.msg import  XXX  #  <<-- replace XXX with the correct type use for cmd t

# TODO: import the LaserScan msg from sensors_msgs package :
from sensor_msgs.msg import XXX

rospy.init_node("TurtleBotPythonNode")
topic_cmd = '/mobile_base/commands/velocity'
cmd_publisher = rospy.Publisher(XXX ,XXX , queue_size=1)


cur_laserscan = LaserScan()
def laserscan_callback(data):
    global cur_laserscan
    cur_laserscan = XXX  # <-- replace XXX here
    # print the current laser scan messag:
    print(XXX)


laser_scan_subscriber = rospy.Subscriber(XXX, XXX, XXX)  #<-- complete here

# define a while loop function that will allows to move the robot forward, and turn if there
while not rospy.is_shutdown():
    # COMPLETE HERE
    # ....
```

Question bonus:

- Run your program on the real turtlebot.

- Add some objects / walls to the simulated environment.