

-실행 준비 작업 pdf 참고 후 ( IP설정을 위해 Nat에서 bridge로 바꿔줘야 가능)

```
#!/usr/bin/env python
```

```
import rospy
```

```
from msg_send.msg import my_msg
```

```
from std_msgs.msg import String
```

```
done = False
```

```
def call_back(data):
```

```
    msg = data.data
```

```
    if ('leong Soo-Rim' in msg):
```

```
        print(msg)
```

```
    else:
```

```
        print("waiting")
```

```
    done = True
```

```
rospy.init_node('remote_student', anonymous=True)
```

```
pub=rospy.Publisher('msg_to_xycar', my_msg, queue_size=1)
```

```
rospy.Subscriber('msg_from_xycar', String, call_back)
```

```
rate = rospy.Rate(0.5)
```

```
msg = my_msg()
```

```
msg.first_name = "Soo-Rim"
```

```
msg.last_name = "leong"
```

```
msg.age = 15
```

```
msg.score = 100
```

```
msg.id_number = 12345678
```

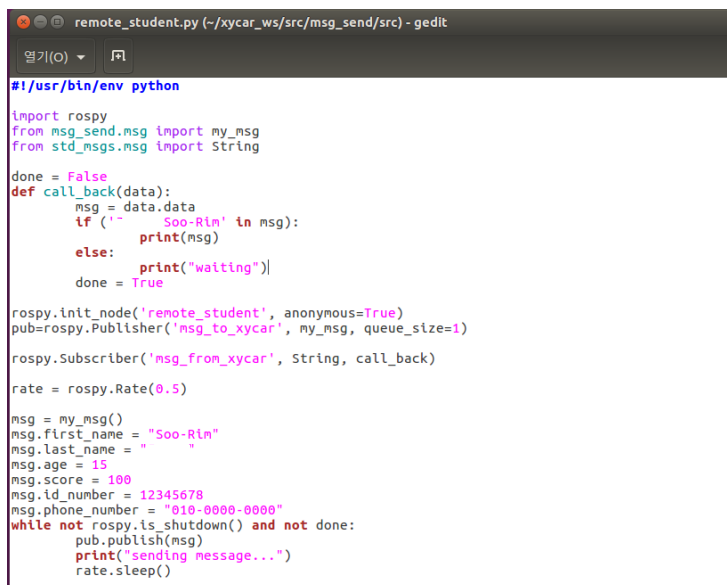
```
msg.phone_number = "010-0000-0000"
```

```
while not rospy.is_shutdown() and not done:
```

```
    pub.publish(msg)
```

```
    print("sending message...")
```

```
    rate.sleep()
```

A screenshot of a code editor window titled 'remote\_student.py (-/xycar\_ws/src/msg\_send/src) - gedit'. The editor shows a Python script for a ROS node. The script imports rospy, msg\_send.msg, my\_msg, and std\_msgs.msg. It defines a callback function 'call\_back' that checks if the received message contains 'Soo-Rim'. If it does, it prints the message; otherwise, it prints 'waiting'. The node is initialized with the name 'remote\_student' and an anonymous=True flag. It publishes to 'msg\_to\_xycar' and subscribes to 'msg\_from\_xycar'. A rate of 0.5 Hz is set. The main loop creates a message object 'msg' with fields: first\_name='Soo-Rim', last\_name='', age=15, score=100, id\_number=12345678, and phone\_number='010-0000-0000'. It then enters a while loop that publishes the message and prints 'sending message...' until the node is shutdown or the 'done' flag is set to True.

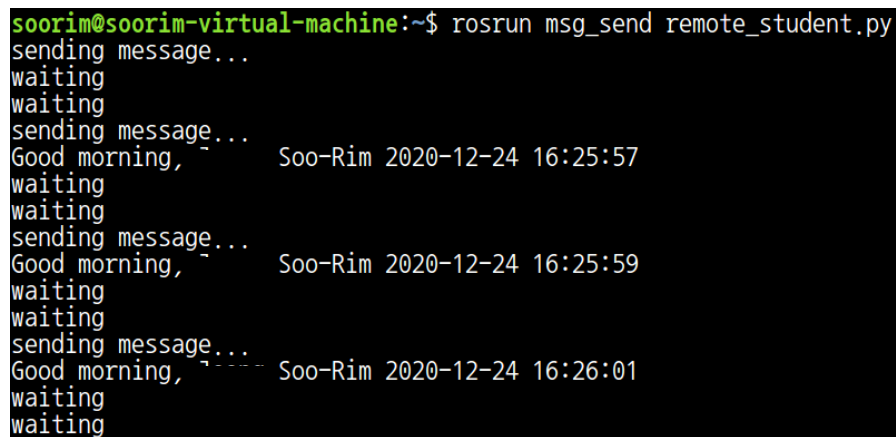
```
#!/usr/bin/env python
import rospy
from msg_send.msg import my_msg
from std_msgs.msg import String

done = False
def call_back(data):
    msg = data.data
    if ('Soo-Rim' in msg):
        print(msg)
    else:
        print("waiting")
    done = True

rospy.init_node('remote_student', anonymous=True)
pub=rospy.Publisher('msg_to_xycar', my_msg, queue_size=1)
rospy.Subscriber('msg_from_xycar', String, call_back)

rate = rospy.Rate(0.5)

msg = my_msg()
msg.first_name = "Soo-Rim"
msg.last_name = ""
msg.age = 15
msg.score = 100
msg.id_number = 12345678
msg.phone_number = "010-0000-0000"
while not rospy.is_shutdown() and not done:
    pub.publish(msg)
    print("sending message...")
    rate.sleep()
```

A screenshot of a terminal window showing the execution of the 'remote\_student.py' script. The prompt is 'soorim@soorim-virtual-machine:~\$'. The command 'roslaunch msg\_send remote\_student.py' is entered. The output shows a sequence of 'sending message...' and 'waiting' messages. When a message is received, it prints 'Good morning, Soo-Rim' followed by a timestamp. The timestamps shown are 2020-12-24 16:25:57, 2020-12-24 16:25:59, and 2020-12-24 16:26:01.

```
soorim@soorim-virtual-machine:~$ roslaunch msg_send remote_student.py
sending message...
waiting
waiting
sending message...
Good morning, Soo-Rim 2020-12-24 16:25:57
waiting
waiting
sending message...
Good morning, Soo-Rim 2020-12-24 16:25:59
waiting
waiting
sending message...
Good morning, Soo-Rim 2020-12-24 16:26:01
waiting
waiting
```