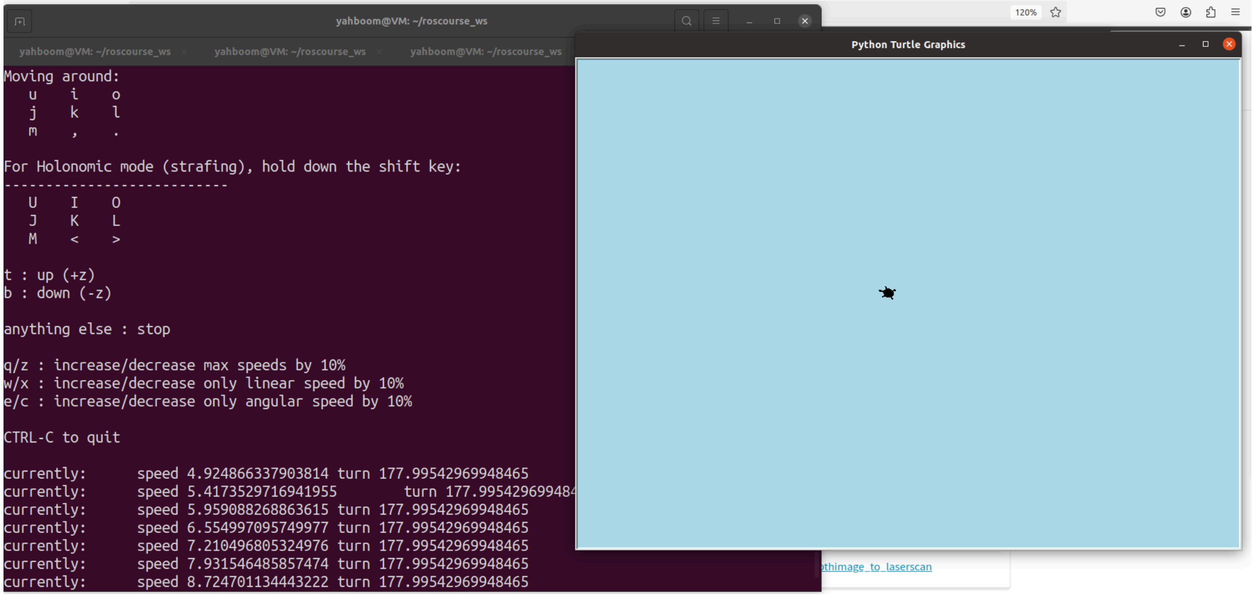
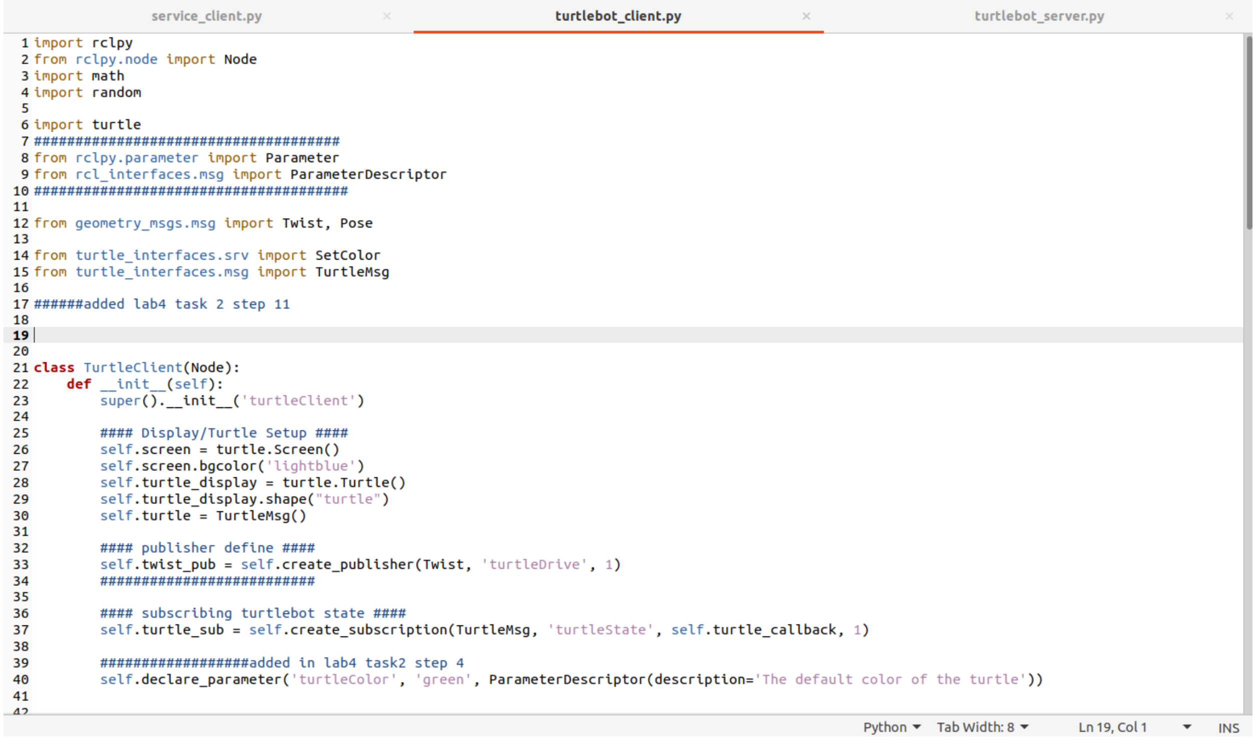
Task1 Step 17:

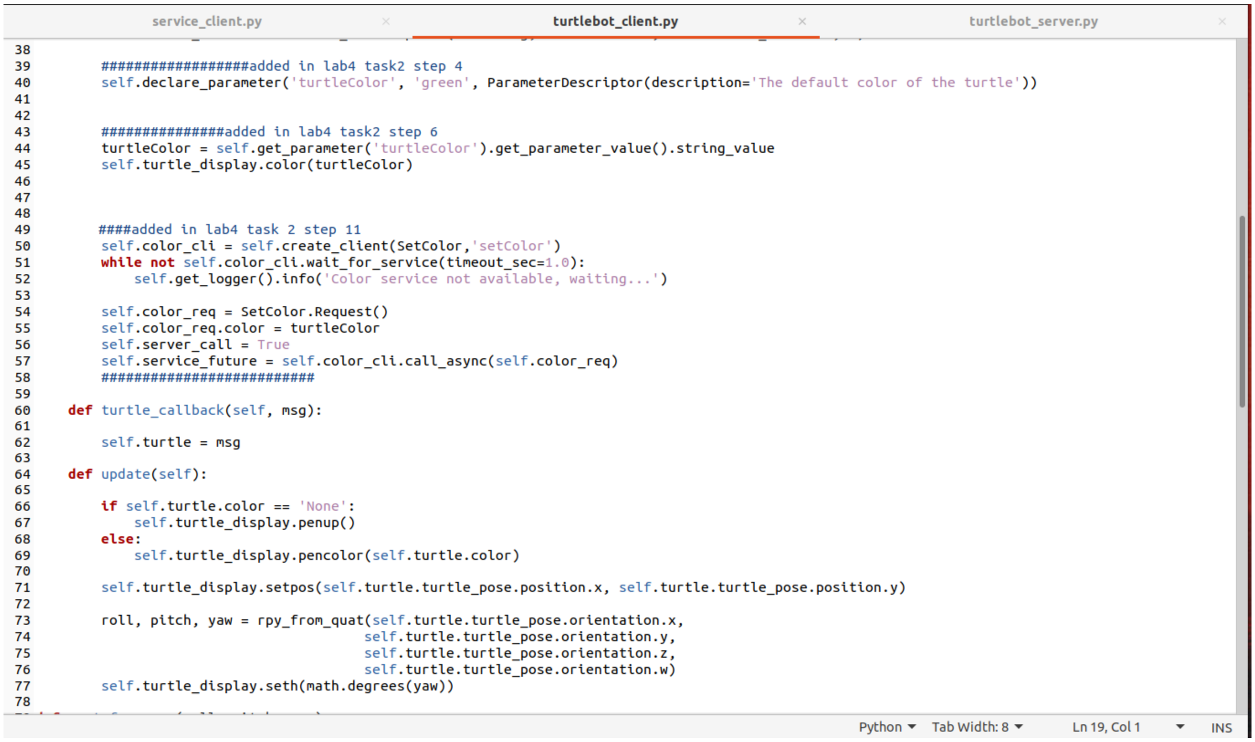

yahboom@VM: ~/roscourse_ws
yahboom@VM: ~/roscourse_ws
yahboom@VM: ~/roscourse_ws
yahboom@VM: ~/roscourse_ws
Python Turtle Graphics
120%
thimage_to_laserscan

```
Moving around:
   u    i    o
   j    k    l
   m    ,    .

For Holonomic mode (strafing), hold down the shift key:
---------------------------
   U    I    O
   J    K    L
   M    <    >

t : up (+z)
b : down (-z)

anything else : stop

q/z : increase/decrease max speeds by 10%
w/x : increase/decrease only linear speed by 10%
e/c : increase/decrease only angular speed by 10%

CTRL-C to quit

currently:      speed 4.924866337903814 turn 177.99542969948465
currently:      speed 5.4173529716941955        turn 177.995429699484
currently:      speed 5.959088268863615 turn 177.99542969948465
currently:      speed 6.554997095749977 turn 177.99542969948465
currently:      speed 7.210496805324976 turn 177.99542969948465
currently:      speed 7.931546485857474 turn 177.99542969948465
currently:      speed 8.724701134443222 turn 177.99542969948465
```

Task2 edited turtlebot_client.py code with comments:

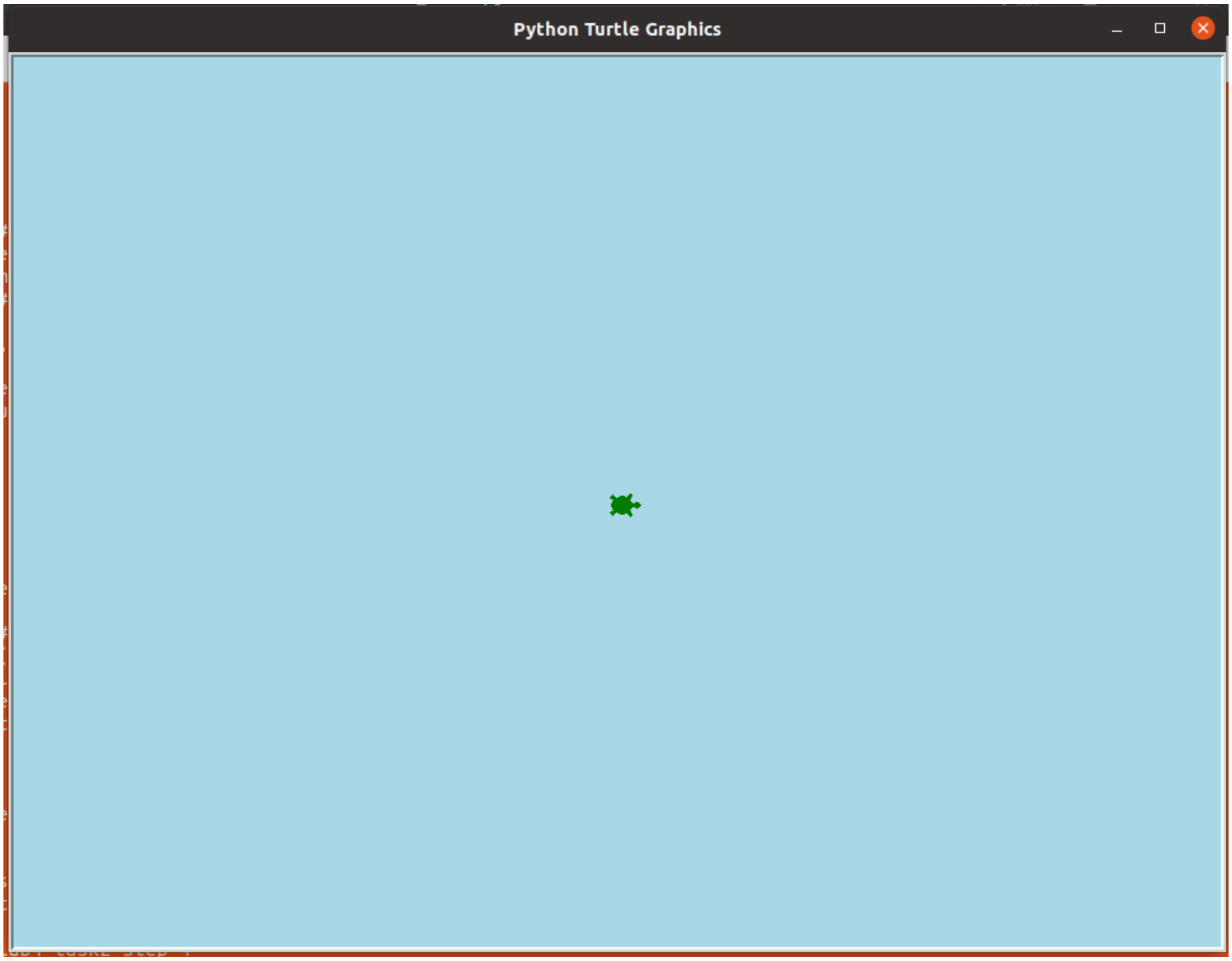


```python
import rclpy
from rclpy.node import Node
import math
import random

import turtle
#####################################
from rclpy.parameter import Parameter
from rcl_interfaces.msg import ParameterDescriptor
######################################

from geometry_msgs.msg import Twist, Pose

from turtle_interfaces.srv import SetColor
from turtle_interfaces.msg import TurtleMsg

######added lab4 task 2 step 11



class TurtleClient(Node):
    def __init__(self):
        super().__init__('turtleClient')

        #### Display/Turtle Setup ####
        self.screen = turtle.Screen()
        self.screen.bgcolor('lightblue')
        self.turtle_display = turtle.Turtle()
        self.turtle_display.shape("turtle")
        self.turtle = TurtleMsg()

        #### publisher define ####
        self.twist_pub = self.create_publisher(Twist, 'turtleDrive', 1)
        ##########################

        #### subscribing turtlebot state ####
        self.turtle_sub = self.create_subscription(TurtleMsg, 'turtleState', self.turtle_callback, 1)

        ##################added in lab4 task2 step 4
        self.declare_parameter('turtleColor', 'green', ParameterDescriptor(description='The default color of the turtle'))

```

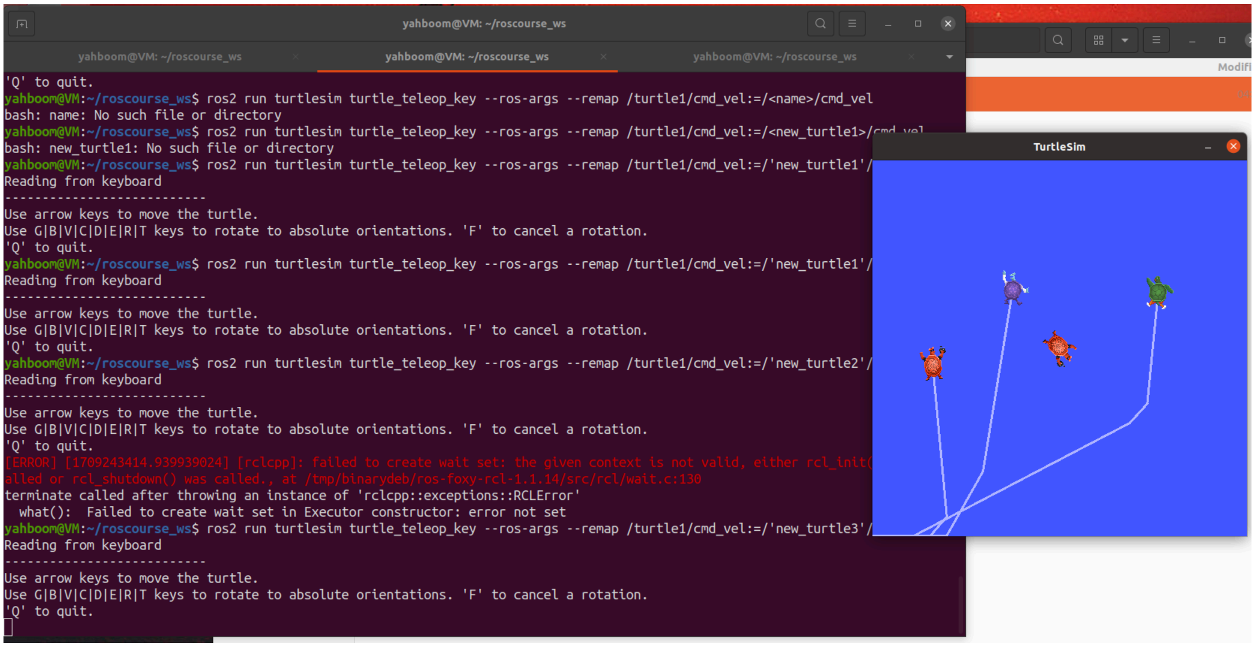


```python

        ##################added in lab4 task2 step 4
        self.declare_parameter('turtleColor', 'green', ParameterDescriptor(description='The default color of the turtle'))


        ###############added in lab4 task2 step 6
        turtleColor = self.get_parameter('turtleColor').get_parameter_value().string_value
        self.turtle_display.color(turtleColor)



        ####added in lab4 task 2 step 11
        self.color_cli = self.create_client(SetColor,'setColor')
        while not self.color_cli.wait_for_service(timeout_sec=1.0):
            self.get_logger().info('Color service not available, waiting...')

        self.color_req = SetColor.Request()
        self.color_req.color = turtleColor
        self.server_call = True
        self.service_future = self.color_cli.call_async(self.color_req)
        ##########################

    def turtle_callback(self, msg):

        self.turtle = msg

    def update(self):

        if self.turtle.color == 'None':
            self.turtle_display.penup()
        else:
            self.turtle_display.pencolor(self.turtle.color)

        self.turtle_display.setpos(self.turtle.turtle_pose.position.x, self.turtle.turtle_pose.position.y)

        roll, pitch, yaw = rpy_from_quat(self.turtle.turtle_pose.orientation.x,
                                         self.turtle.turtle_pose.orientation.y,
                                         self.turtle.turtle_pose.orientation.z,
                                         self.turtle.turtle_pose.orientation.w)
        self.turtle_display.seth(math.degrees(yaw))

```

Task2 Step 14:
A green turtle



Task4 Step3:


yahboom@VM: ~/roscourse_ws
yahboom@VM: ~/roscourse_ws
yahboom@VM: ~/roscourse_ws
yahboom@VM: ~/roscourse_ws
TurtleSim
Modifi

```
'Q' to quit.
yahboom@VM:~/roscourse_ws$ ros2 run turtlesim turtle_teleop_key --ros-args --remap /turtle1/cmd_vel:=/<name>/cmd_vel
bash: name: No such file or directory
yahboom@VM:~/roscourse_ws$ ros2 run turtlesim turtle_teleop_key --ros-args --remap /turtle1/cmd_vel:=/<new_turtle1>/cmd_vel
bash: new_turtle1: No such file or directory
yahboom@VM:~/roscourse_ws$ ros2 run turtlesim turtle_teleop_key --ros-args --remap /turtle1/cmd_vel:=/'new_turtle1'/
Reading from keyboard
---------------------------
Use arrow keys to move the turtle.
Use G|B|V|C|D|E|R|T keys to rotate to absolute orientations. 'F' to cancel a rotation.
'Q' to quit.
yahboom@VM:~/roscourse_ws$ ros2 run turtlesim turtle_teleop_key --ros-args --remap /turtle1/cmd_vel:=/'new_turtle1'/
Reading from keyboard
---------------------------
Use arrow keys to move the turtle.
Use G|B|V|C|D|E|R|T keys to rotate to absolute orientations. 'F' to cancel a rotation.
'Q' to quit.
yahboom@VM:~/roscourse_ws$ ros2 run turtlesim turtle_teleop_key --ros-args --remap /turtle1/cmd_vel:=/'new_turtle2'/
Reading from keyboard
---------------------------
Use arrow keys to move the turtle.
Use G|B|V|C|D|E|R|T keys to rotate to absolute orientations. 'F' to cancel a rotation.
'Q' to quit.
[ERROR] [1709243414.939939024] [rclcpp]: failed to create wait set: the given context is not valid, either rcl_init(
alled or rcl_shutdown() was called., at /tmp/binarydeb/ros-foxy-rcl-1.1.14/src/rcl/wait.c:130
terminate called after throwing an instance of 'rclcpp::exceptions::RCLError'
  what():  Failed to create wait set in Executor constructor: error not set
yahboom@VM:~/roscourse_ws$ ros2 run turtlesim turtle_teleop_key --ros-args --remap /turtle1/cmd_vel:=/'new_turtle3'/
Reading from keyboard
---------------------------
Use arrow keys to move the turtle.
Use G|B|V|C|D|E|R|T keys to rotate to absolute orientations. 'F' to cancel a rotation.
'Q' to quit.
```