```python
#!/usr/bin/env python3

#TODO 0 Update the header below with your information
#Junhyung Park and Ryan Buckton
#mouse_client_OO.py
#ROS Enabled mouse client
#Activates when mouse wheel is scrolled down and
#Deactivates when mouse wheel is scrolled up
#last modified 2 Feb 2023

import rospy
import pyautogui
import sys
import time
import os
from pynput.mouse import Listener
# TODO 1 Import the lab1 message file you created for the Mouse Controller data
from lab1.msg import MouseController

#Initialize global variable that will keep track of mouse button status
global activate
activate = False


#get dimensions of screen so that we can scale for (-1,1) in both x and y
xDim,yDim = pyautogui.size()

class Mouse:
    def __init__(self):
        global activate
        listener = Listener(on_move=self.on_move, on_click=self.on_click,
on_scroll=self.on_scroll)
        listener.start()
        self.ctrl_c = False
        rospy.on_shutdown(self.shutdownhook)

        # TODO 2 Create class variables to store activation status, and x and y
position
        # Initialize the controller status to False, and in the center of the screen
        self.msg = MouseController()
        self.msg.status.data = False
        self.msg.xPos = 0.0
        self.msg.yPos = 0.0

        # TODO 3 Create a publisher to publish MouseController data on the mouse_info
topic
        self.pub = rospy.Publisher('mouse_info', MouseController, queue_size = 1)


    # Handle the event of the user moving the mouse
    def on_move(self,x,y):
        #We need to establish the use of the global variable for button activation
        global activate

        if(activate == False):
```

```
            messageString = "No movement sent, activate cursor first by scrolling down
on the mouse wheel"
            print(messageString, end='')
            # overwrite same line
            print('\b' *len(messageString), end='', flush=True)
            xScale = 0.0
            yScale = 0.0
        else:
            # Use the dimensions of the monitor to convert output into x & y between
-1 and 1
            xScale = ((x/(xDim/2))-1)
            yScale = -((y/(yDim/2))-1)
            xFormat = "{:+.3f}".format(xScale)
            yFormat = "{:+.3f}".format(yScale)
            # create 19 byte message with x,y coordinates embedded and print to screen
            positionStr = 'X: ' + str(xFormat) + ' Y: ' + str(yFormat) + "
"
            print(positionStr, end='')
            # overwrite same line
            print('\b' *len(positionStr), end='', flush=True)
        # TODO 4 Update the appropriate class variables and publish
        self.msg.xPos = xScale
        self.msg.yPos = yScale
        self.pub.publish(self.msg)

    #Currently I'm not doing anything with the button click.
    #Eventually we could add additional modes if we wanted to.
    def on_click(self,x,y,button,pressed):
        pass

    # Handle the event of the user using the scroll wheel on the mouse
    def on_scroll(self,x,y,dx,dy):
        global activate
        if dy == 1:
            os.system("clear")
            print("Welcome to the mouse controller!\n\n")
            print("In order to ACTIVATE the controller scroll down on the mouse
wheel.\n\n")
            print("In order to DEACTIVATE the controller scroll up on the mouse
wheel.\n\n")
            print("To close the program, left-click on the terminal window and type
ctrl-c.\n\n")
            activate = False
        if dy == -1:
            activate = True
        # TODO 5 Update the appropriate class variables and publish
        self.msg.status.data = activate
        self.pub.publish(self.msg)


    # Handle the event of the user pressing ctrl_c
    def shutdownhook(self):
        # TODO 6 Update the appropriate class variables and publish before shutdown
        self.msg.xPos = 0.00
        self.msg.yPos = 0.00
        self.msg.status.data = False
```

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

        print("Shutting down the client
")
        self.ctrl_c = True



if __name__ == "__main__":
    os.system("clear")
    print("Welcome to the mouse controller!\n\n")
    print("In order to ACTIVATE the controller scroll down on the mouse wheel.\n\n")
    print("In order to DEACTIVATE the controller scroll up on the mouse wheel.\n\n")
    print("To close the program, left-click on the terminal window and type ctrl-
c.\n\n")
    rospy.init_node('Mouse')
    Mouse()
    rospy.spin()
```