Digital Instrument Control Using Computer Programming

Sejin Jeon
Sogang University Physics Department
Student ID 20231262

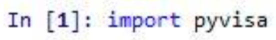
(6th Week Post-Experiment Lab Report)

I. PRACTICE 1

A. Writing Code in the Console Tab

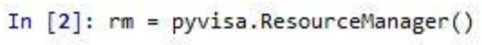
In the first practice, we were asked to use the programs provided to use the AFG2021 (function generator). In the first part of the first practice, we were asked to simply insert the code in the console tab and observe what happens. There were a total of 14 lines of code to be tested.

1. The first code provided simply imported the PyVISA package, a Python package that enables measurement instrument control.



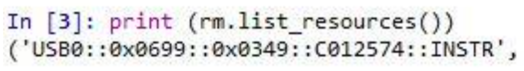
```
In [1]: import pyvisa
```

2. The second code assigns the resource manager class in the PyVISA package to the variable ”rm”.



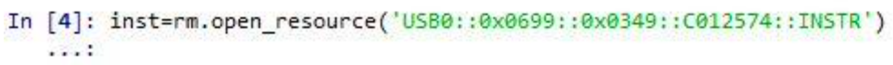
```
In [2]: rm = pyvisa.ResourceManager()
```

3. The third code uses the ”list resources” function in the resource manager class, printing out all possible addresses for the instrument that we want to control.



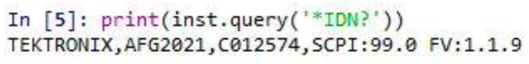
```
In [3]: print (rm.list_resources())
('USB0::0x0699::0x0349::C012574::INSTR',
```

4. The fourth and fifth code allows us to identify the address of the digital instrument input. The fourth code assigns the ”open resource” function (of the resource manager class) with the input being the computer’s connected devices to the variable ”inst”.



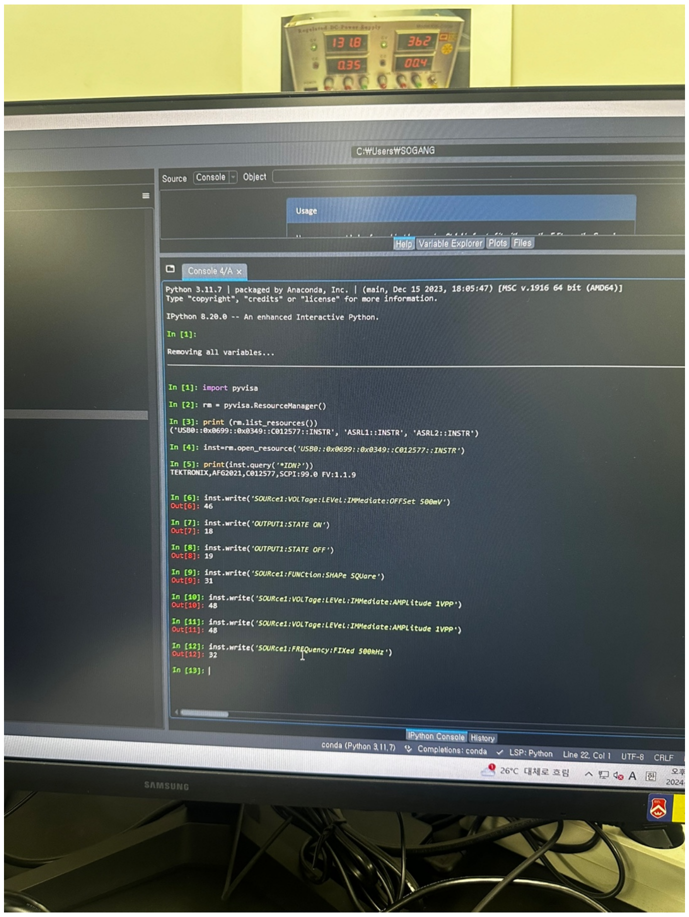
```
In [4]: inst=rm.open_resource('USB0::0x0699::0x0349::C012574::INSTR')
   ...:
```

5. The fifth code uses the the ”query” function, which takes the opened resource lists that we created above and gives the identification of these devices. The particular device with the identification ”AFG2021” is the one we want to use as the VISA address for the AFG2021 function generator.



```
In [5]: print(inst.query('*IDN?'))
TEKTRONIX,AFG2021,C012574,SCPI:99.0 FV:1.1.9
```

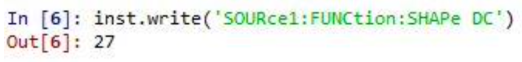
The implementation of the codes above in the actual practice process looked like the following.



```
Python 3.11.7 | packaged by Anaconda, Inc. | (main, Dec 15 2023, 18:05:47) [MSC v.1916 64 bit (AMD64)]
Type "copyright", "credits" or "license" for more information.

IPython 8.20.0 -- An enhanced Interactive Python.

In [1]:

Removing all variables...

In [1]: import pyvisa

In [2]: rm = pyvisa.ResourceManager()

In [3]: print (rm.list_resources())
('USB0::0x0699::0x0349::C012577::INSTR', 'ASRL1::INSTR', 'ASRL2::INSTR')

In [4]: inst=rm.open_resource('USB0::0x0699::0x0349::C012577::INSTR')

In [5]: print(inst.query('*IDN?'))
TEKTRONIX,AFG2021,C012577,SCPI:99.0 FV:1.1.9


In [6]: inst.write('SOURce1:VOLTage:LEVel:IMMediate:OFFSet 500mV')
Out[6]: 46

In [7]: inst.write('OUTPUT1:STATE ON')
Out[7]: 18

In [8]: inst.write('OUTPUT1:STATE OFF')
Out[8]: 19

In [9]: inst.write('SOURce1:FUNCtion:SHAPe SQuare')
Out[9]: 31

In [10]: inst.write('SOURce1:VOLTage:LEVel:IMMediate:AMPLitude 1VPP')
Out[10]: 48

In [11]: inst.write('SOURce1:VOLTage:LEVel:IMMediate:AMPLitude 1VPP')
Out[11]: 48

In [12]: inst.write('SOURce1:FREQuency:FIXed 500kHz')
Out[12]: 32

In [13]: |
```

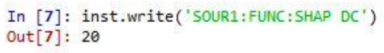
FIG. 1: Code used in practice.

6. From the 6th code line, we controlled the instrument, observing the effects of each code on the instrument. The 6th code line of the listing changed the function shape of the function generator to a DC output, like the following.



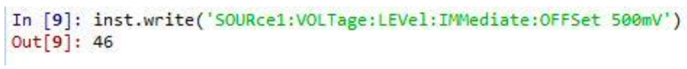
```
In [6]: inst.write('SOURce1:FUNCtion:SHAPe DC')
Out[6]: 27
```

7. The 7th code line had the same effect of the code above.

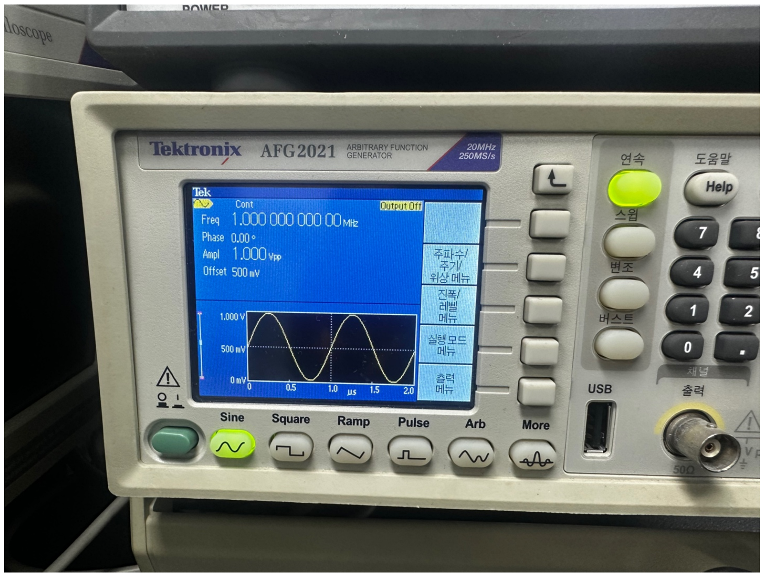


```
In [7]: inst.write('SOUR1:FUNC:SHAP DC')
Out[7]: 20
```

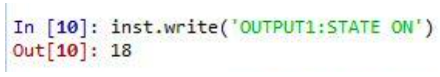
8. The 8th code line set the offset level of the first source, resulting in the function generator to show an interface like the following.



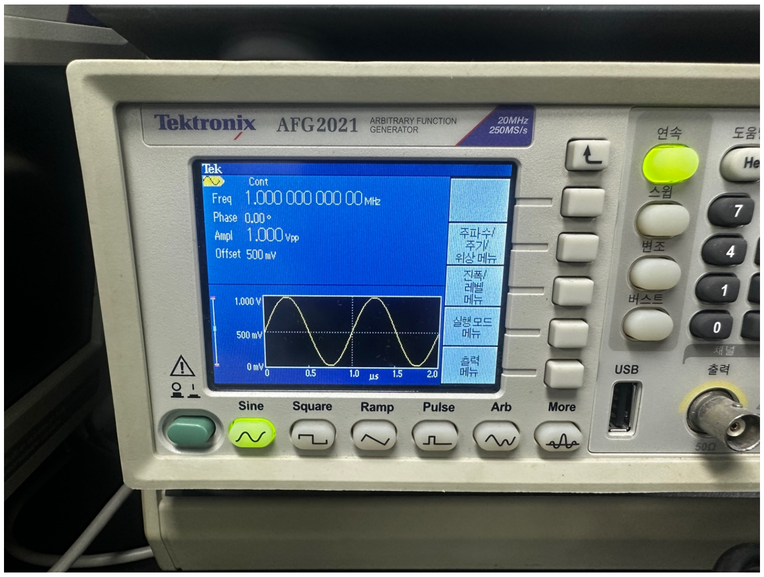
```
In [9]: inst.write('SOURce1:VOLTage:LEVel:IMMediate:OFFSet 500mV')
Out[9]: 46
```



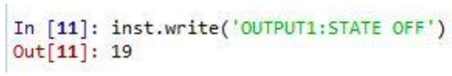
9. The 9th code line turned the first output on, resulting in the function generator to show an interface like the following.



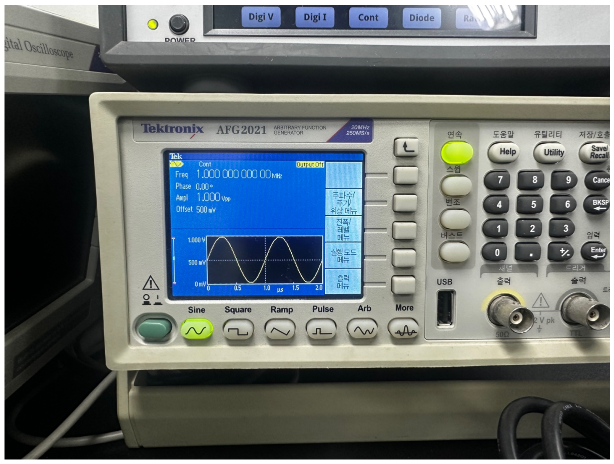
```
In [10]: inst.write('OUTPUT1:STATE ON')
Out[10]: 18
```



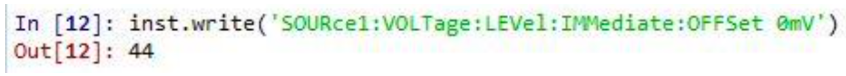
10. The 10th line of code did the opposite of the 9th line, turning the output off.



```
In [11]: inst.write('OUTPUT1:STATE OFF')
Out[11]: 19
```

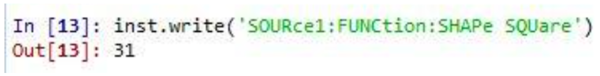


11. The 11th code line set the offset level of the first source to 0 mV.

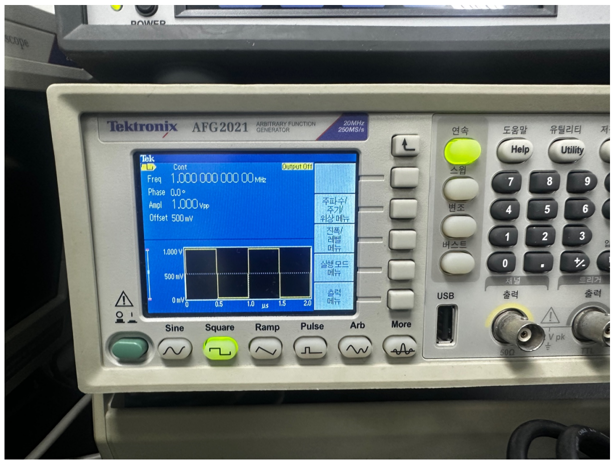


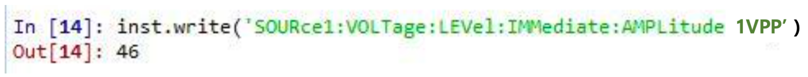
```
In [12]: inst.write('SOURce1:VOLTage:LEVel:IMMediate:OFFSet 0mV')
Out[12]: 44
```

12. The 12th line of code creating the made the function generator to generate a function shape of a square, implemented as follows.

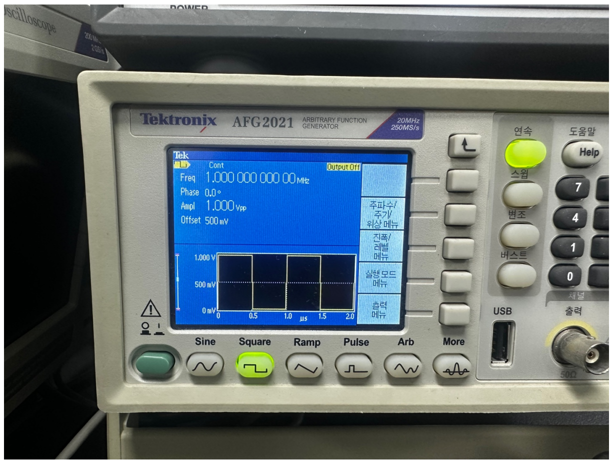


```
In [13]: inst.write('SOURce1:FUNCtion:SHAPe SQUare')
Out[13]: 31
```

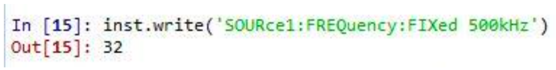




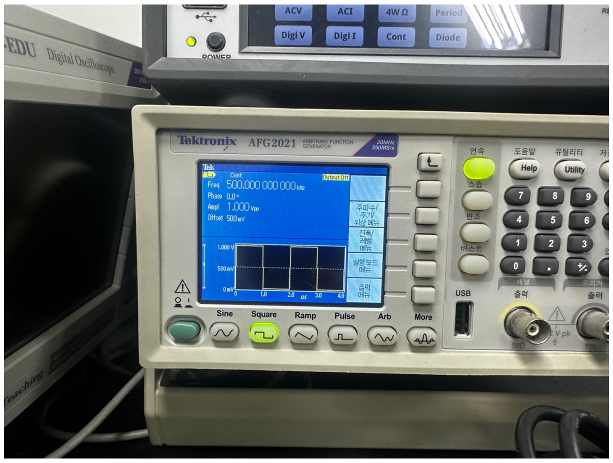
```
In [14]: inst.write('SOURce1:VOLTage:LEVel:IMMediate:AMPLitude 1VPP')
Out[14]: 46
```



13. The 13th line of code made the amplitude of the signal to have a peak-to-peak value of 1 V, implemented as follows.
14. The 14th line of code creating the made the function generator to generate a function of a frequency of a fixed 500 kHz.

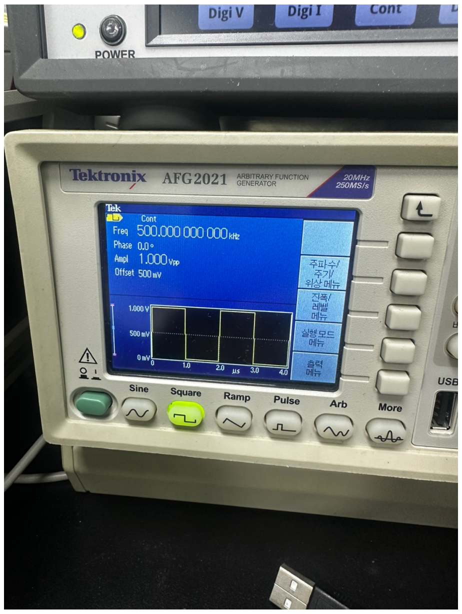


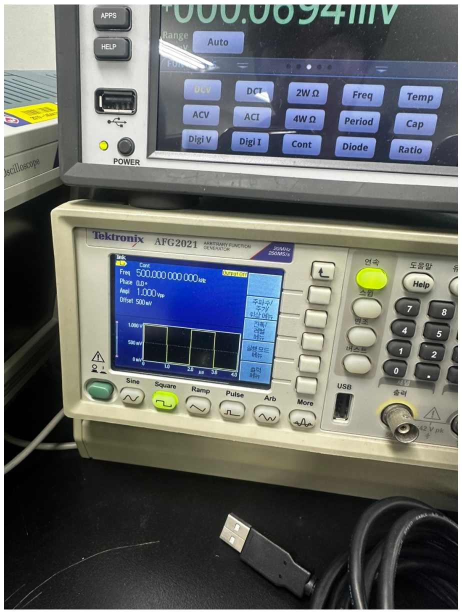
```
In [15]: inst.write('SOURce1:FREQuency:FIXed 500kHz')
Out[15]: 32
```

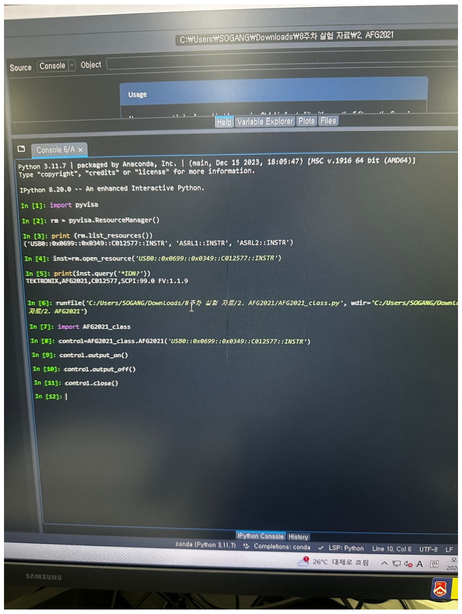


B. Defining a Class and Using a Generated Object

Using the file "AFG2021_class.py", a certain class "AFG2021" was used to turn the function generator on and off. The results can be seen below.



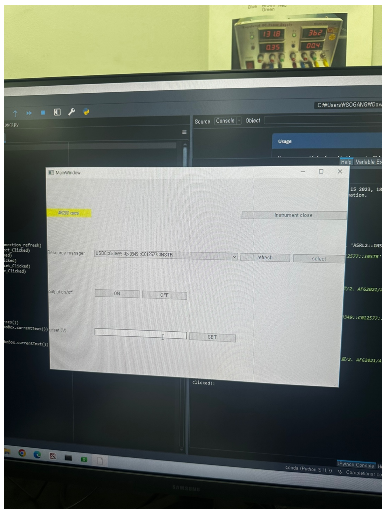


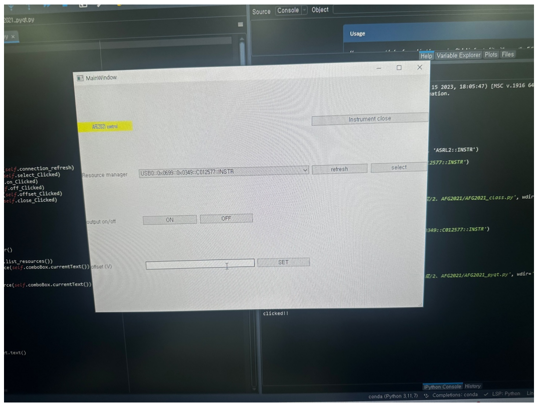


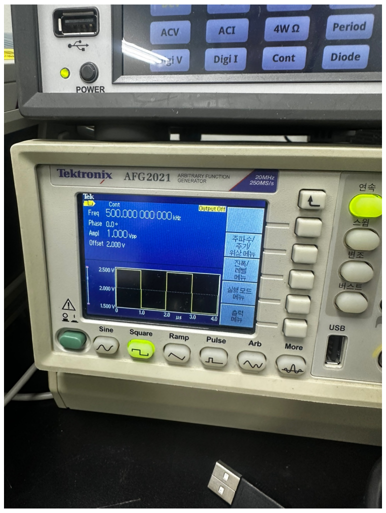
```
Python 3.11.7 | packaged by Anaconda, Inc. | (main, Dec 15 2023, 18:05:47) [MSC v.1916 64 bit (AMD64)]
Type "copyright", "credits" or "license" for more information.

IPython 8.20.0 -- An enhanced Interactive Python.

In [1]: import pyvisa

In [2]: rm = pyvisa.ResourceManager()

In [3]: print (rm.list_resources())
('USB0::0x0699::0x0349::C012577::INSTR', 'ASRL1::INSTR', 'ASRL2::INSTR')

In [4]: inst=rm.open_resource('USB0::0x0699::0x0349::C012577::INSTR')

In [5]: print(inst.query('*IDN?'))
TEKTRONIX,AFG2021,C012577,SCPI:99.0 FV:1.1.9


In [6]: runfile('C:/Users/SOGANG/Downloads/8주차 실험 자료/2. AFG2021/AFG2021_class.py', wdir='C:/Users/SOGANG/Downlo
자료/2. AFG2021')

In [7]: import AFG2021_class

In [8]: control=AFG2021_class.AFG2021('USB0::0x0699::0x0349::C012577::INSTR')

In [9]: control.output_on()

In [10]: control.output_off()

In [11]: control.close()

In [12]: 
```

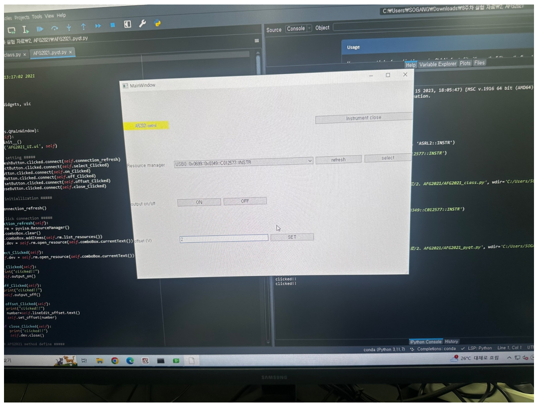
C. Creating a Graphical User Interface (GUI) Using *PyQt* and *Qt Designer*

In the third part of the first practice, we used the code in the files "AFG2021_UI" and "AFG2021_pyqt.py" to create a GUI interface.





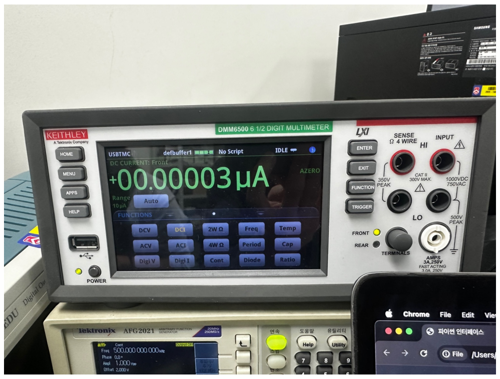


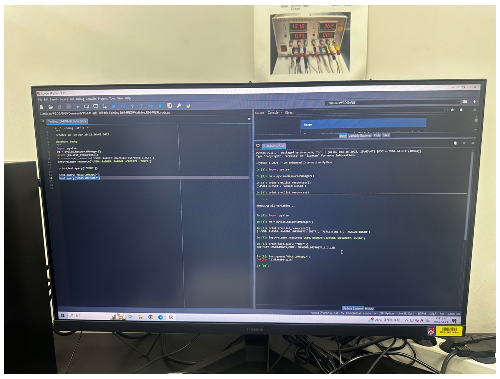


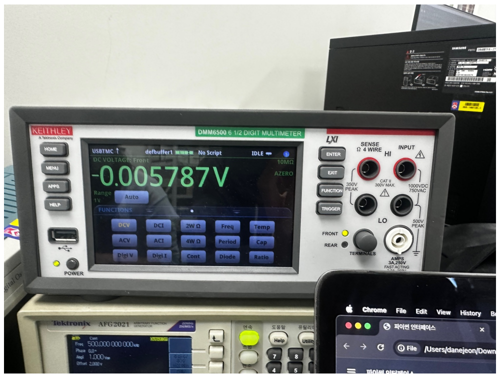
II. PRACTICE 2

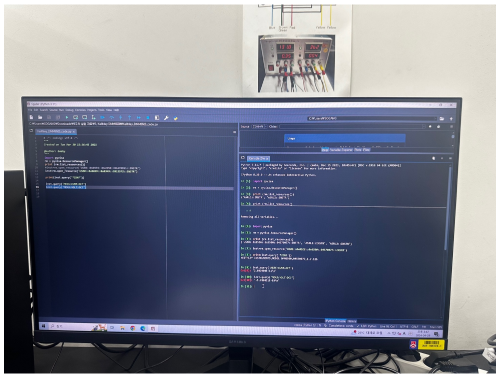
A. Writing Code in the Console Tab

In the second practice, the same instructions for the practice above was repeated, where the lines of the code let us measure the amplitude and the voltage using the Keithley DMM6500 (multimeter).



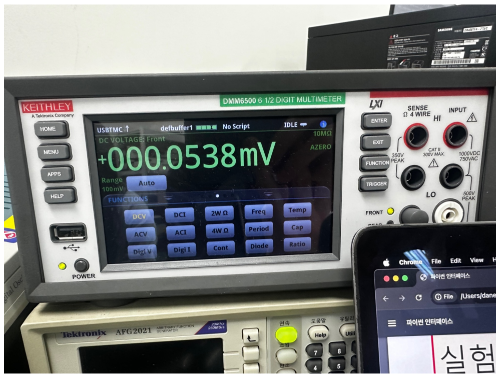


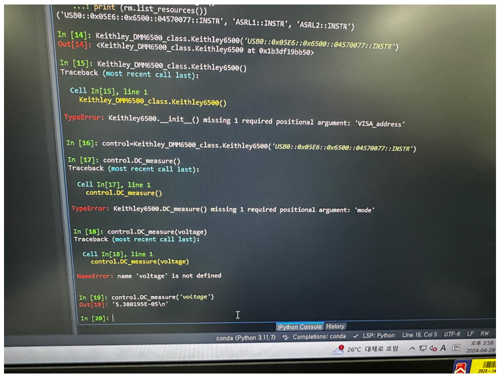




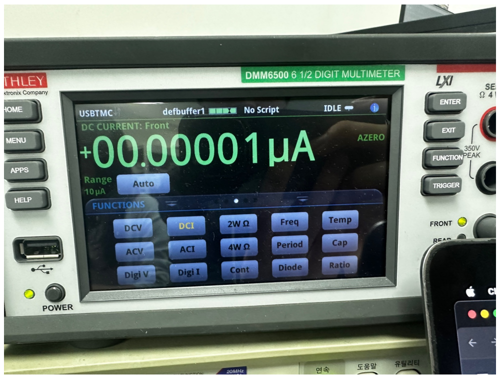
B. Defining a Class and Using a Generated Object

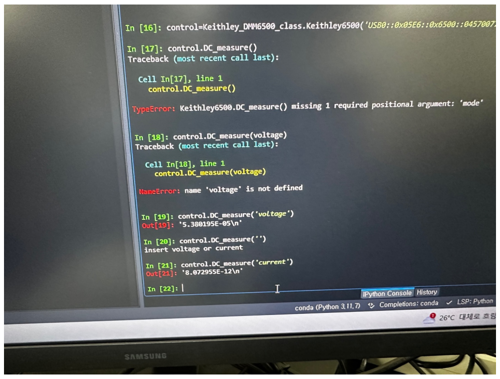
After the first lines of code above, the file ”Keithley_DMM6500_class.py” was used to change the mode of the measurements. The ”if” argument was successfully used.





```
   ...: print (rm.list_resources())
('USB0::0x05E6::0x6500::04570077::INSTR', 'ASRL1::INSTR', 'ASRL2::INSTR')

In [14]: Keithley_DMM6500_class.Keithley6500('USB0::0x05E6::0x6500::04570077::INSTR')
Out[14]: <Keithley_DMM6500_class.Keithley6500 at 0x1b3df19bb50>

In [15]: Keithley_DMM6500_class.Keithley6500()
Traceback (most recent call last):

  Cell In[15], line 1
    Keithley_DMM6500_class.Keithley6500()

TypeError: Keithley6500.__init__() missing 1 required positional argument: 'VISA_address'


In [16]: control=Keithley_DMM6500_class.Keithley6500('USB0::0x05E6::0x6500::04570077::INSTR')

In [17]: control.DC_measure()
Traceback (most recent call last):

  Cell In[17], line 1
    control.DC_measure()

TypeError: Keithley6500.DC_measure() missing 1 required positional argument: 'mode'


In [18]: control.DC_measure(voltage)
Traceback (most recent call last):

  Cell In[18], line 1
    control.DC_measure(voltage)

NameError: name 'voltage' is not defined


In [19]: control.DC_measure('voltage')
Out[19]: '5.380195E-05\n'

In [20]: |
```





```
In [16]: control=Keithley_DMM6500_class.Keithley6500('USB0::0x05E6::0x6500::0457007

In [17]: control.DC_measure()
Traceback (most recent call last):

  Cell In[17], line 1
    control.DC_measure()

TypeError: Keithley6500.DC_measure() missing 1 required positional argument: 'mode'


In [18]: control.DC_measure(voltage)
Traceback (most recent call last):

  Cell In[18], line 1
    control.DC_measure(voltage)

NameError: name 'voltage' is not defined


In [19]: control.DC_measure('voltage')
Out[19]: '5.380195E-05\n'

In [20]: control.DC_measure('')
insert voltage or current

In [21]: control.DC_measure('current')
Out[21]: '8.072955E-12\n'

In [22]: |
```

[1] The Sogang University Physics Department. Experimental physics 1 manual. *“Creating a Communications and Remote Control Environment (Python 1)”*.