

2022년 IoT기반 스마트 솔루션 개발자 양성과정



# Programming : Python

## 19-Connect to Arduino

담당 교수 : 윤 종 이

010-9577-1696

[ojo1696@naver.com](mailto:ojo1696@naver.com)

<https://cafe.naver.com/yoons2022>



충북대학교 공동훈련센터

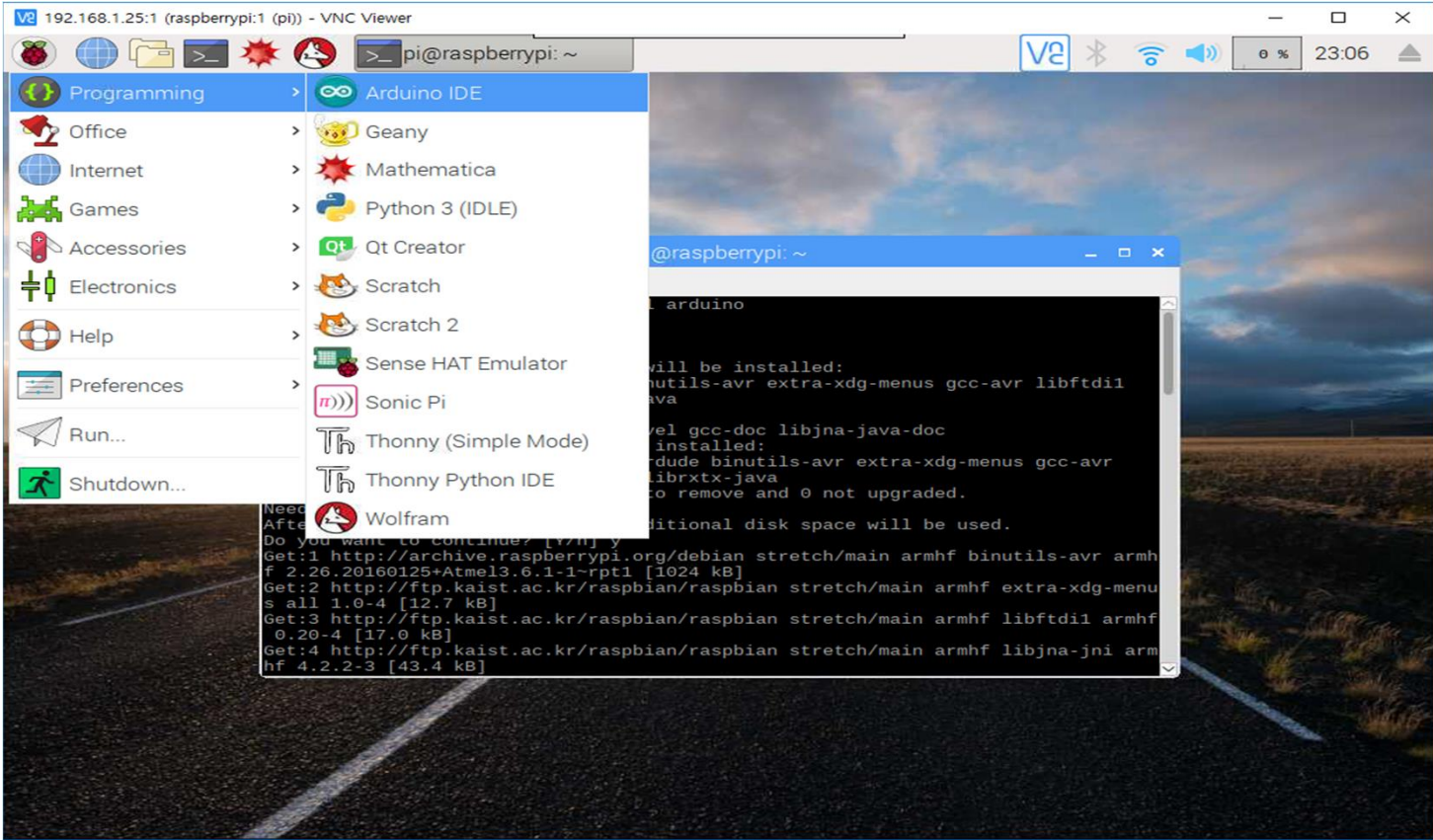
# Install Arduino

- \$ sudo apt-get install arduino

```
pi@raspberrypi: ~  
File Edit Tabs Help  
pi@raspberrypi:~ $ sudo apt install arduino  
Reading package lists... Done  
Building dependency tree  
Reading state information... Done  
The following additional packages will be installed:  
  arduino-core avr-libc avrdude binutils-avr extra-xdg-menus gcc-avr libftdi1  
  libjna-java libjna-jni librxxtx-java  
Suggested packages:  
  arduino-mk avrdude-doc task-c-devel gcc-doc libjna-java-doc  
The following NEW packages will be installed:  
  arduino arduino-core avr-libc avrdude binutils-avr extra-xdg-menus gcc-avr  
  libftdi1 libjna-java libjna-jni librxxtx-java  
0 upgraded, 11 newly installed, 0 to remove and 0 not upgraded.  
Need to get 21.6 MB of archives.  
After this operation, 132 MB of additional disk space will be used.  
Do you want to continue? [Y/n] y  
Get:1 http://archive.raspberrypi.org/debian stretch/main armhf binutils-avr armhf  
  2.26.20160125+Atmel3.6.1-1~rpt1 [1024 kB]  
Get:2 http://ftp.kaist.ac.kr/raspbian/raspbian stretch/main armhf extra-xdg-menu  
  s all 1.0-4 [12.7 kB]  
Get:3 http://ftp.kaist.ac.kr/raspbian/raspbian stretch/main armhf libftdi1 armhf  
  0.20-4 [17.0 kB]  
Get:4 http://ftp.kaist.ac.kr/raspbian/raspbian stretch/main armhf libjna-jni arm  
  hf 4.2.2-3 [43.4 kB]
```

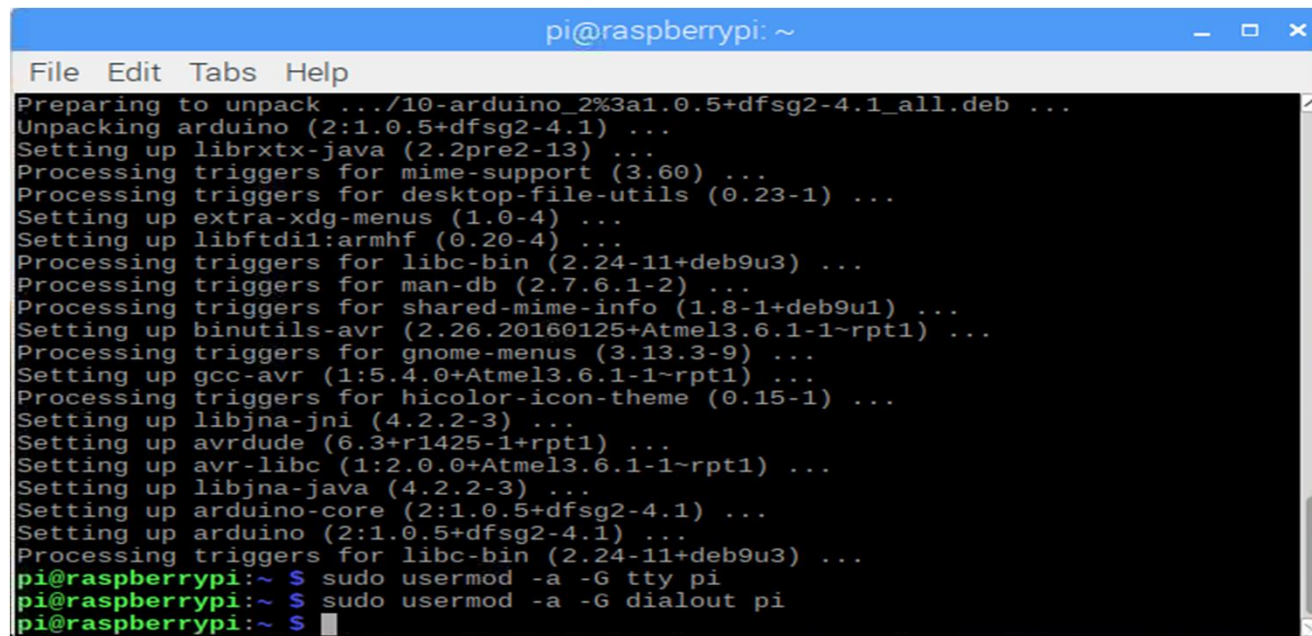


# Arduino IDE



# Set Serial Port

- \$ sudo usermod -a -G tty pi
- \$ sudo usermod -a -G dialout pi



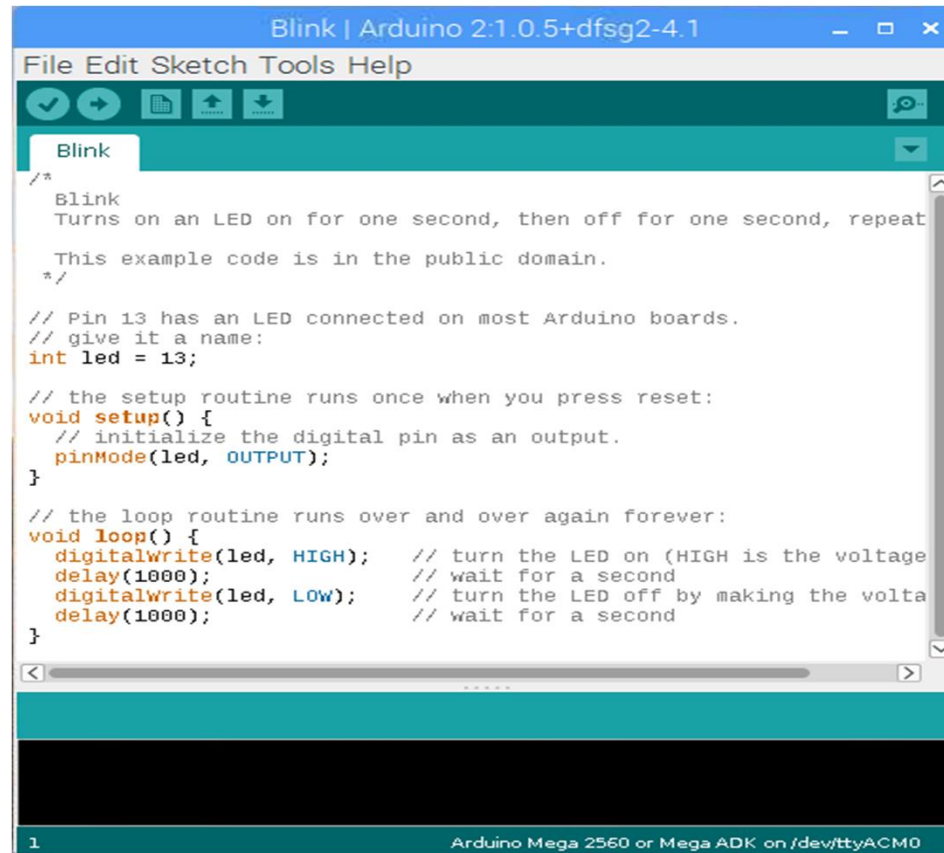
```
pi@raspberrypi: ~  
File Edit Tabs Help  
Preparing to unpack .../10-arduino_2%3a1.0.5+dfsg2-4.1_all.deb ...  
Unpacking arduino (2:1.0.5+dfsg2-4.1) ...  
Setting up librx-tx-java (2.2pre2-13) ...  
Processing triggers for mime-support (3.60) ...  
Processing triggers for desktop-file-utils (0.23-1) ...  
Setting up extra-xdg-menus (1.0-4) ...  
Setting up libftdi1:armhf (0.20-4) ...  
Processing triggers for libc-bin (2.24-11+deb9u3) ...  
Processing triggers for man-db (2.7.6.1-2) ...  
Processing triggers for shared-mime-info (1.8-1+deb9u1) ...  
Setting up binutils-avr (2.26.20160125+Atmel3.6.1-1~rpt1) ...  
Processing triggers for gnome-menus (3.13.3-9) ...  
Setting up gcc-avr (1:5.4.0+Atmel3.6.1-1~rpt1) ...  
Processing triggers for hicolor-icon-theme (0.15-1) ...  
Setting up libjna-jni (4.2.2-3) ...  
Setting up avrdude (6.3+r1425-1+rpt1) ...  
Setting up avr-libc (1:2.0.0+Atmel3.6.1-1~rpt1) ...  
Setting up libjna-java (4.2.2-3) ...  
Setting up arduino-core (2:1.0.5+dfsg2-4.1) ...  
Setting up arduino (2:1.0.5+dfsg2-4.1) ...  
Processing triggers for libc-bin (2.24-11+deb9u3) ...  
pi@raspberrypi:~$ sudo usermod -a -G tty pi  
pi@raspberrypi:~$ sudo usermod -a -G dialout pi  
pi@raspberrypi:~$
```

Serial Port : /dev/ttyACM0



충북대학교 공동훈련센터

# Blink

A screenshot of the Arduino IDE interface. The title bar reads "Blink | Arduino 2:1.0.5+dfsg2-4.1". The menu bar includes "File", "Edit", "Sketch", "Tools", and "Help". Below the menu bar is a toolbar with icons for opening, saving, uploading, and downloading. The main text area contains the following code:

```
/*
 * Blink
 * Turns on an LED on for one second, then off for one second, repeat
 *
 * This example code is in the public domain.
 */

// Pin 13 has an LED connected on most Arduino boards.
// give it a name:
int led = 13;

// the setup routine runs once when you press reset:
void setup() {
  // initialize the digital pin as an output.
  pinMode(led, OUTPUT);
}

// the loop routine runs over and over again forever:
void loop() {
  digitalWrite(led, HIGH); // turn the LED on (HIGH is the voltage
  delay(1000);             // wait for a second
  digitalWrite(led, LOW);  // turn the LED off by making the volta
  delay(1000);             // wait for a second
}
```

The bottom status bar shows "1" on the left and "Arduino Mega 2560 or Mega ADK on /dev/ttyACM0" on the right.

# Serial Transmit

- 0~9 까지의 숫자를 1초간격으로 전송

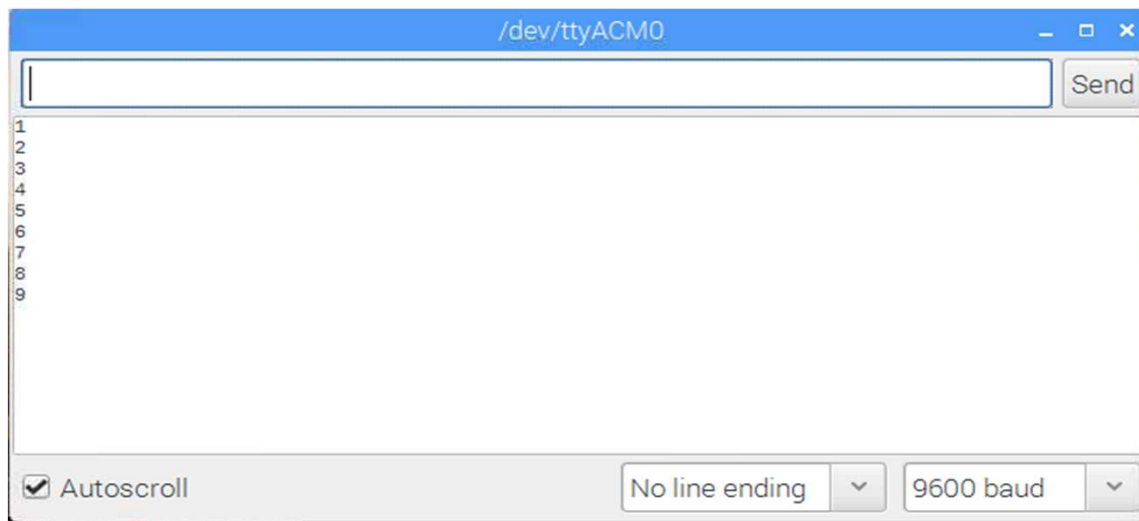
```
int Count=0;

void setup() {
  Serial.begin(9600);
}

void loop() {
  if (++Count>9) Count=0;
  Serial.println(Count);
  delay(1000);
}
```



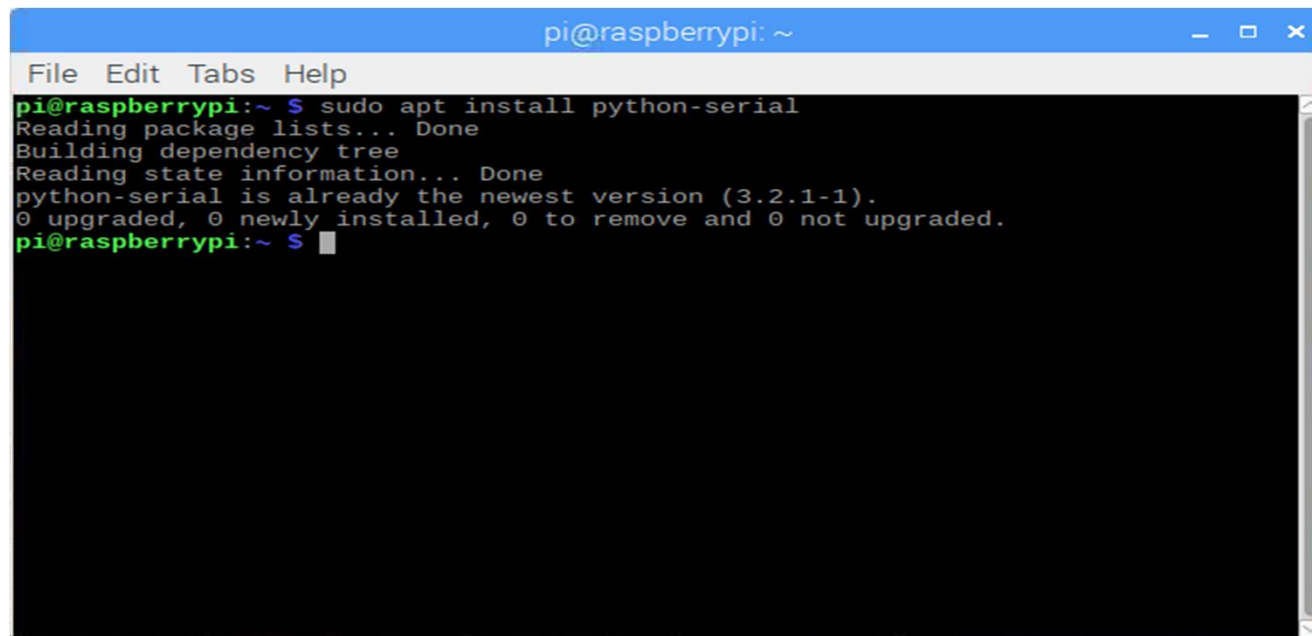
# Serial Monitor





# Install Python-Serial

- `$ sudo apt install python-serial`

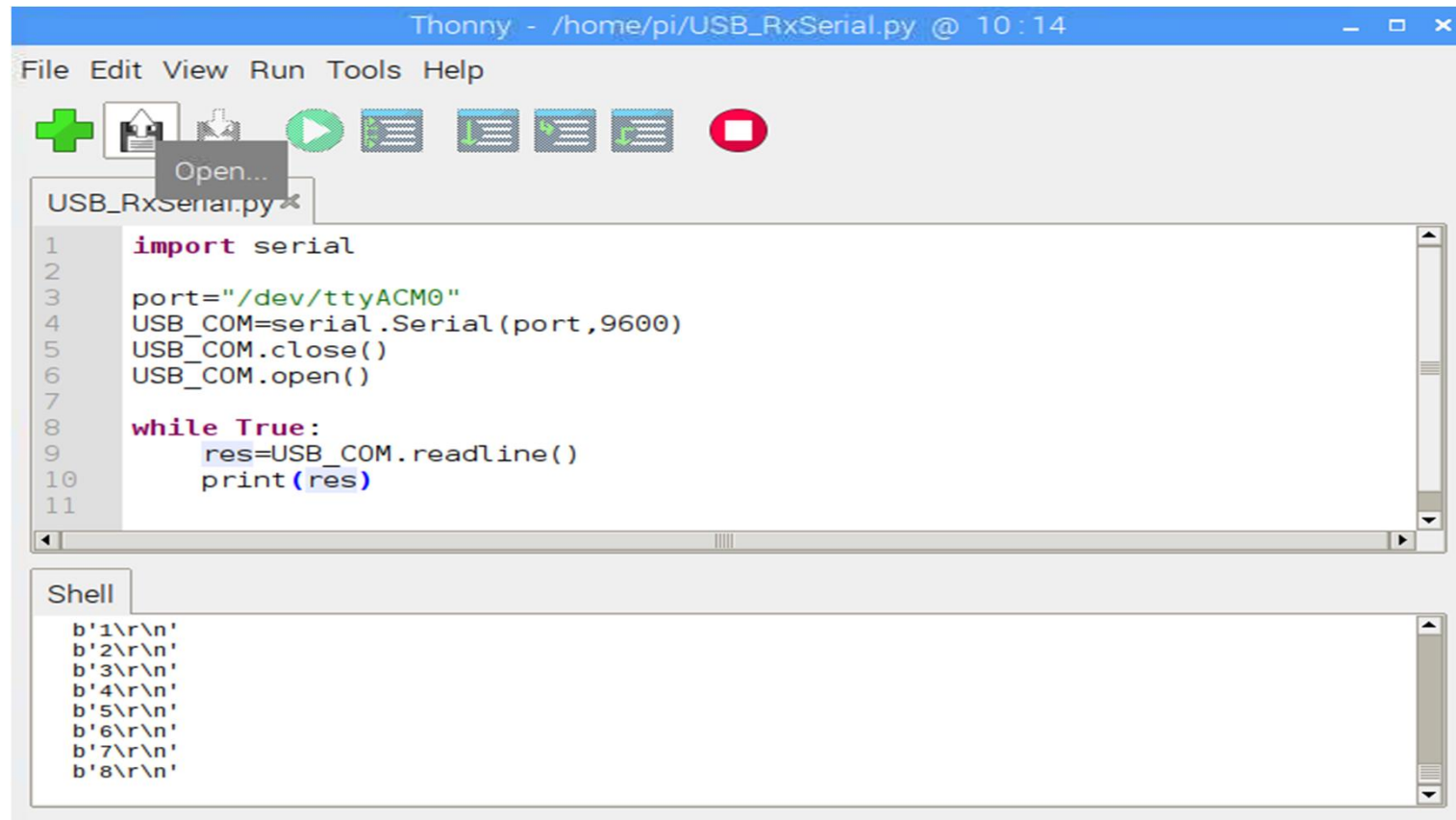


```
pi@raspberrypi: ~  
File Edit Tabs Help  
pi@raspberrypi:~ $ sudo apt install python-serial  
Reading package lists... Done  
Building dependency tree  
Reading state information... Done  
python-serial is already the newest version (3.2.1-1).  
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.  
pi@raspberrypi:~ $
```





# USB\_RxSerial.py



Thonny - /home/pi/USB\_RxSerial.py @ 10:14

File Edit View Run Tools Help

Open...

USB\_RxSerial.py

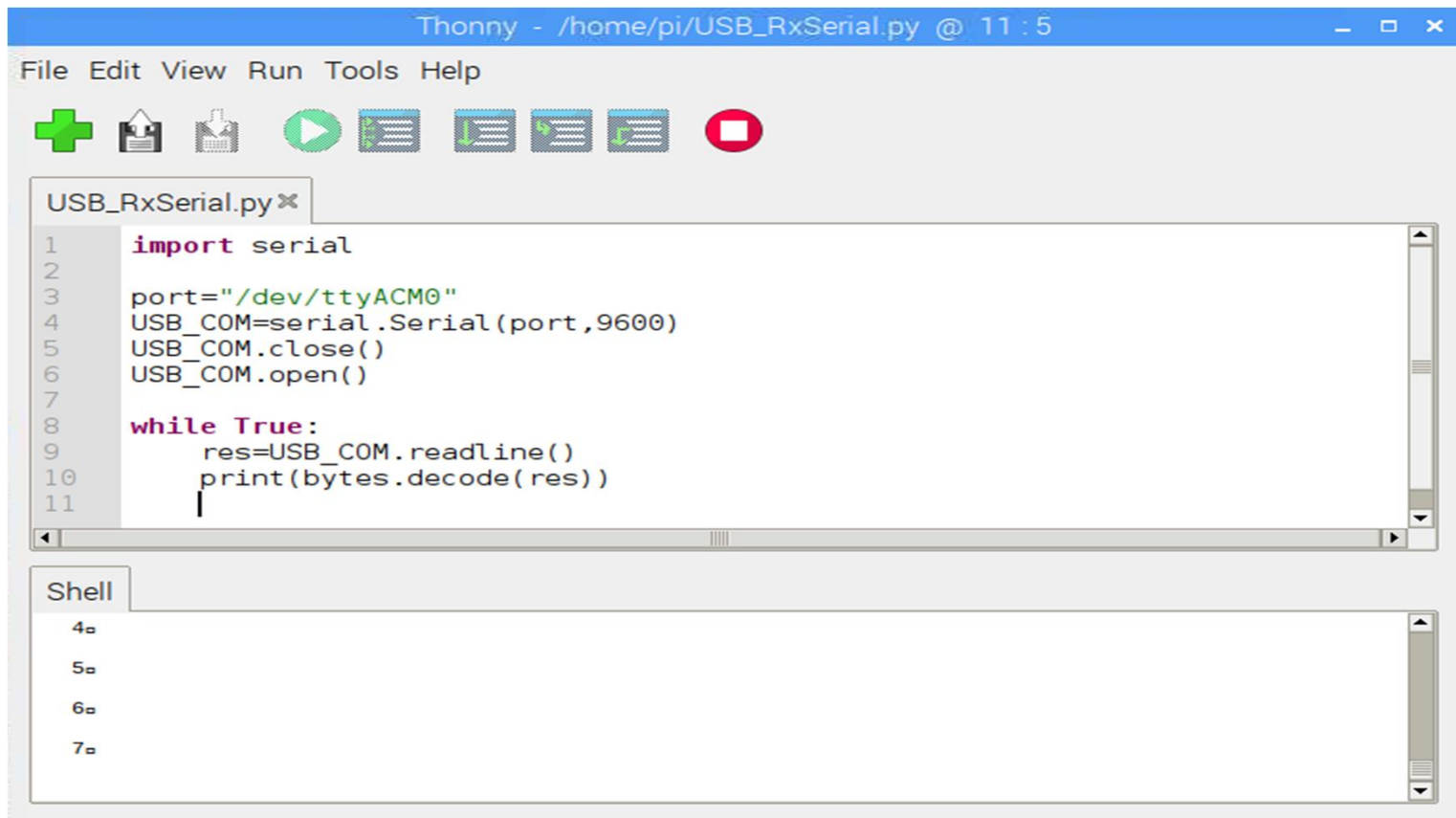
```
1 import serial
2
3 port="/dev/ttyACM0"
4 USB_COM=serial.Serial(port,9600)
5 USB_COM.close()
6 USB_COM.open()
7
8 while True:
9     res=USB_COM.readline()
10    print(res)
11
```

Shell

```
b'1\r\n'
b'2\r\n'
b'3\r\n'
b'4\r\n'
b'5\r\n'
b'6\r\n'
b'7\r\n'
b'8\r\n'
```



# Bytes.decode( )



The screenshot shows the Thonny IDE interface. The title bar reads "Thonny - /home/pi/USB\_RxSerial.py @ 11:5". The menu bar includes "File", "Edit", "View", "Run", "Tools", and "Help". Below the menu is a toolbar with icons for file operations and execution. The main editor window displays the file "USB\_RxSerial.py" with the following Python code:

```
1 import serial
2
3 port="/dev/ttyACM0"
4 USB_COM=serial.Serial(port,9600)
5 USB_COM.close()
6 USB_COM.open()
7
8 while True:
9     res=USB_COM.readline()
10    print(bytes.decode(res))
11
```

Below the editor is a "Shell" window showing the output of the program:

```
4a
5a
6a
7a
```



# Serial4.py

```
import serial
from tkinter import *

port="/dev/ttyACM0"
USB_COM=serial.Serial(port,9600)
if (USB_COM.is_open):
    USB_COM.close( )
USB_COM.open( )

window=Tk( )
window.title("Serial Monitor")

scrollbar=Scrollbar(window)
scrollbar.pack(side=RIGHT,fill=Y)
```

```
log = Text(window, width=30,height=30, takefocus=0)
log.pack( )

log.config(yscrollcommand=scrollbar.set)
scrollbar.config(command=log.yview)

serBuffer=""

def readSerial( ):
    res=USB_COM.readline( )
    log.insert('0.0',res)
    window.after(10, readSerial)

window.after(1000, readSerial)
window.mainloop( )
```



# Run Serial4.py

