Duncan Jones

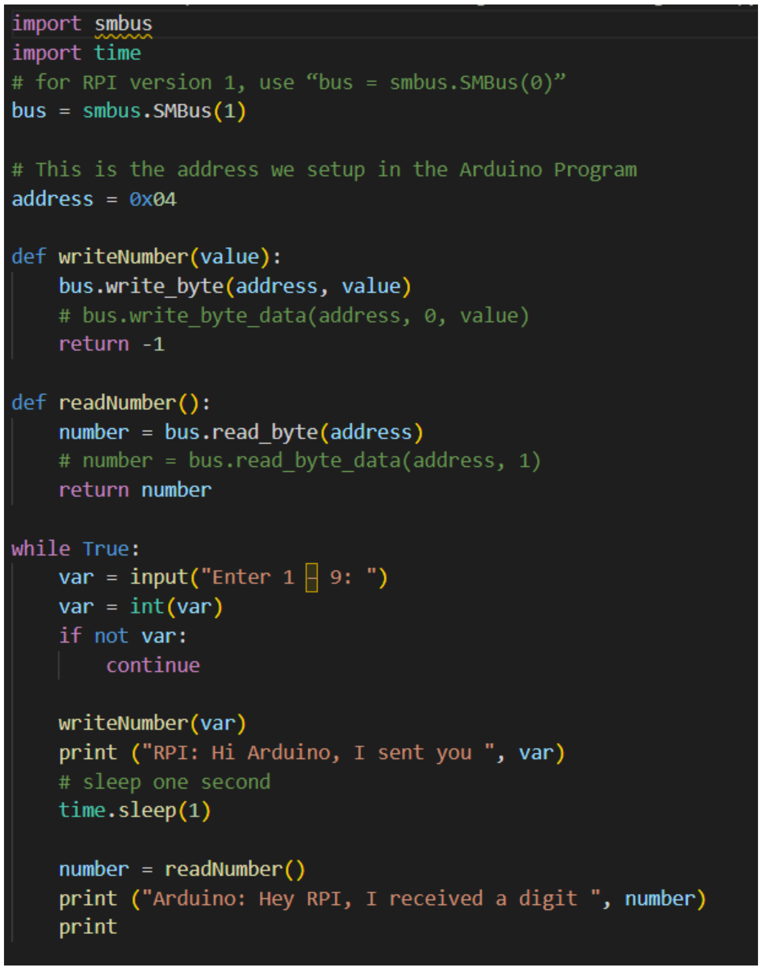
September 23, 2022

Assignment 2 Documentation

1. Yes, data can be transferred up to 3.4Mb/s based off what mode you are using. In standard mode data can be transmitted up to 100Kb/s, fast mode up to 1Mb/s, and high-speed mode which allows for up to 3.4Mb/s.
2. The max rate of data transfer is 3.4Mb/s
3. Wire Library
4. 32 bits

Code

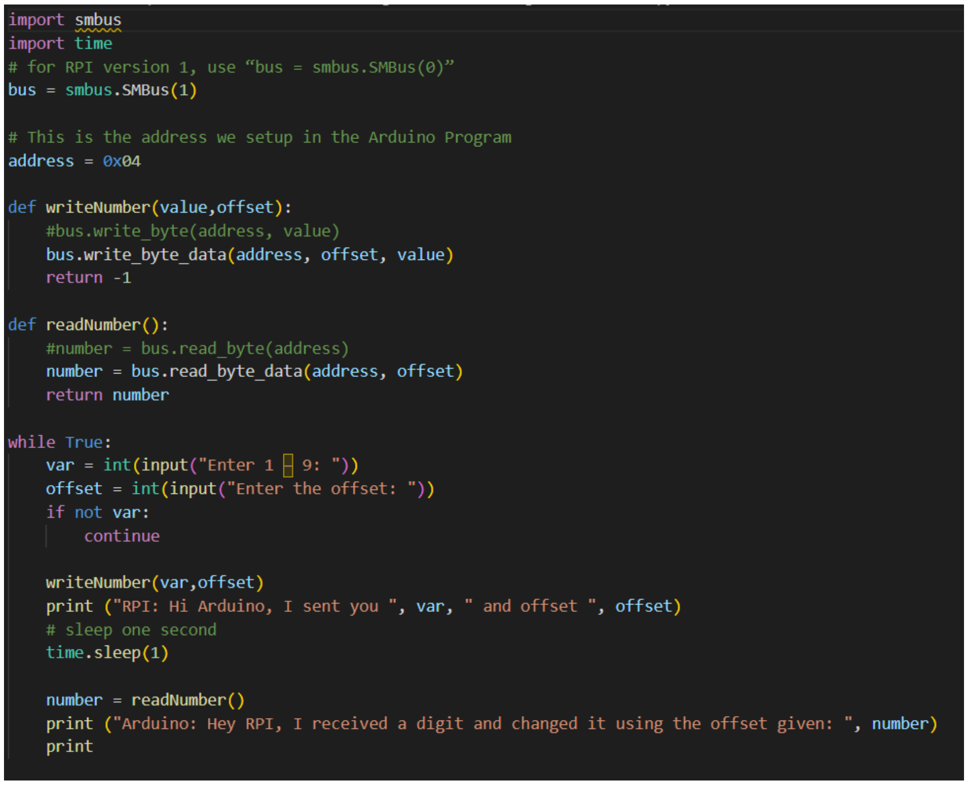
Exercise One:



```python
import smbus
import time
# for RPI version 1, use “bus = smbus.SMBus(0)”
bus = smbus.SMBus(1)

# This is the address we setup in the Arduino Program
address = 0x04

def writeNumber(value):
    bus.write_byte(address, value)
    # bus.write_byte_data(address, 0, value)
    return -1

def readNumber():
    number = bus.read_byte(address)
    # number = bus.read_byte_data(address, 1)
    return number

while True:
    var = input("Enter 1 ▯ 9: ")
    var = int(var)
    if not var:
        continue

    writeNumber(var)
    print ("RPI: Hi Arduino, I sent you ", var)
    # sleep one second
    time.sleep(1)

    number = readNumber()
    print ("Arduino: Hey RPI, I received a digit ", number)
    print
```

```
#include <Wire.h>

#define SLAVE_ADDRESS 0x04
int number = 0;
int state = 0;

void setup() {
  pinMode(13, OUTPUT);
  Serial.begin(9600); // start serial for output
  // initialize i2c as slave
  Wire.begin(SLAVE_ADDRESS);

  // define callbacks for i2c communication
  Wire.onReceive(receiveData);
  Wire.onRequest(sendData);

  Serial.println("Ready!");
}

void loop() {
  delay(100);
}

// callback for received data
void receiveData(int byteCount){

  while(Wire.available()) {
    number = Wire.read();
    Serial.print("data received: ");

   Serial.println(number);

   if (number == 1){

      if (state == 0){
       digitalWrite(13, HIGH); // set the LED on
       state = 1;
      }
     else{
       digitalWrite(13, LOW); // set the LED off
       state = 0;
     }
   }
  }
}

// callback for sending data
void sendData(){
  number = number + 5 ;
  Wire.write(number);
}
```

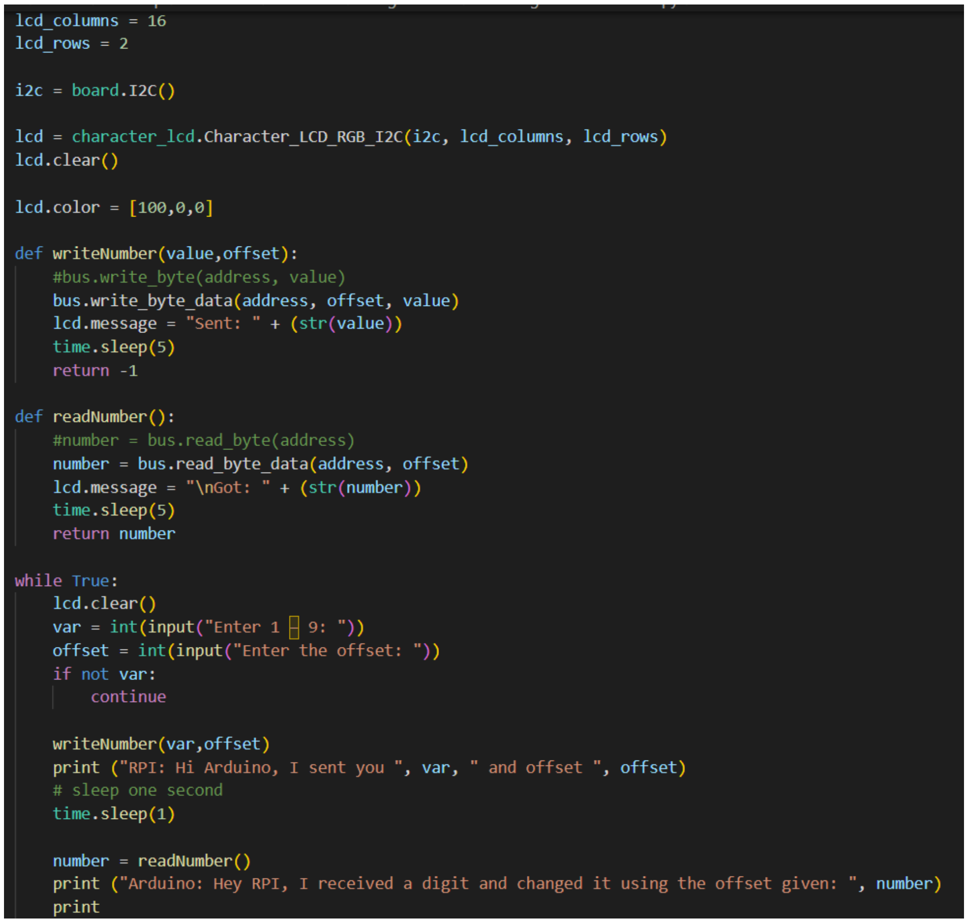
Exercise 2



```python
import smbus
import time
# for RPI version 1, use “bus = smbus.SMBus(0)”
bus = smbus.SMBus(1)

# This is the address we setup in the Arduino Program
address = 0x04

def writeNumber(value,offset):
    #bus.write_byte(address, value)
    bus.write_byte_data(address, offset, value)
    return -1

def readNumber():
    #number = bus.read_byte(address)
    number = bus.read_byte_data(address, offset)
    return number

while True:
    var = int(input("Enter 1 – 9: "))
    offset = int(input("Enter the offset: "))
    if not var:
        continue

    writeNumber(var,offset)
    print ("RPI: Hi Arduino, I sent you ", var, " and offset ", offset)
    # sleep one second
    time.sleep(1)

    number = readNumber()
    print ("Arduino: Hey RPI, I received a digit and changed it using the offset given: ", number)
    print
```

```
#include <Wire.h>

#define SLAVE_ADDRESS 0x04
int number = 0;
int number1 = 0 ;
int state = 0;
int data[32] = {0} ;

void setup() {
  pinMode(13, OUTPUT);
  Serial.begin(9600); // start serial for output
  // initialize i2c as slave
  Wire.begin(SLAVE_ADDRESS);

  // define callbacks for i2c communication
  Wire.onReceive(receiveData);
  Wire.onRequest(sendData);

  Serial.println("Ready!");
}

void loop() {
  delay(100);
}

// callback for received data
void receiveData(int byteCount){
  int i = 0 ;
  while(Wire.available()) {
    Serial.print(data[i]) ;
    Serial.print(" ") ;
    i++ ;



  }
  i-- ;
  Serial.println(" ") ;
}

// callback for sending data
void sendData(){
  if (data[1] != 0) {
    if(data[0] == 0) {
      number = data[1] + 5 ;
      Wire.write(number) ;
    }
    else if (data[0] == 1) {
      number1 = data[1] + 10 ;
      Wire.write(number1) ;
    }
  }
  else {
    Wire.write(data[0] + 5) ;
  }
  number = number + 5 ;
  Wire.write(number);
}
```

Exercise 3

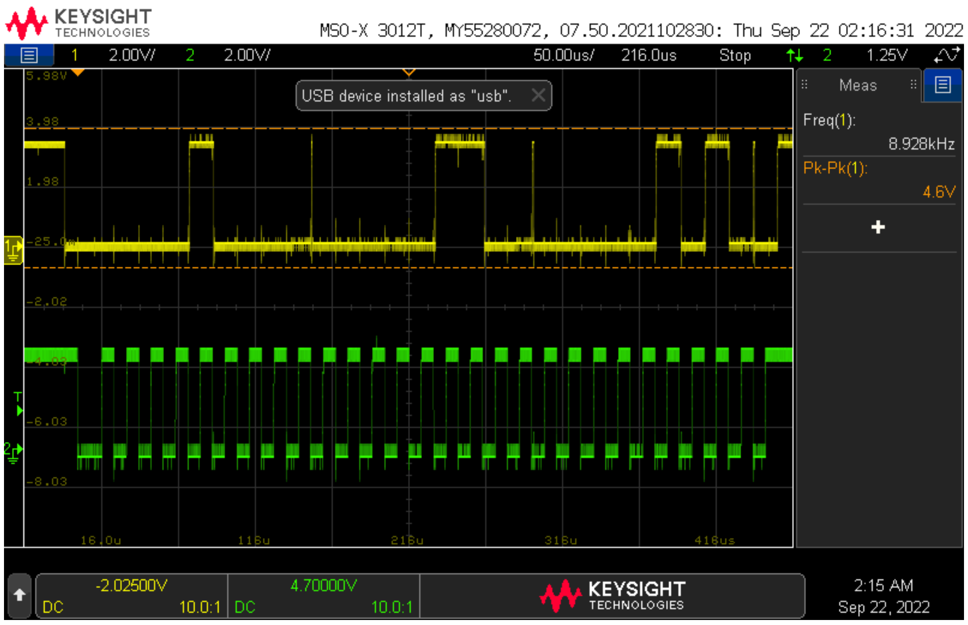


```python
lcd_columns = 16
lcd_rows = 2

i2c = board.I2C()

lcd = character_lcd.Character_LCD_RGB_I2C(i2c, lcd_columns, lcd_rows)
lcd.clear()

lcd.color = [100,0,0]

def writeNumber(value,offset):
    #bus.write_byte(address, value)
    bus.write_byte_data(address, offset, value)
    lcd.message = "Sent: " + (str(value))
    time.sleep(5)
    return -1

def readNumber():
    #number = bus.read_byte(address)
    number = bus.read_byte_data(address, offset)
    lcd.message = "\nGot: " + (str(number))
    time.sleep(5)
    return number

while True:
    lcd.clear()
    var = int(input("Enter 1 - 9: "))
    offset = int(input("Enter the offset: "))
    if not var:
        continue

    writeNumber(var,offset)
    print ("RPI: Hi Arduino, I sent you ", var, " and offset ", offset)
    # sleep one second
    time.sleep(1)

    number = readNumber()
    print ("Arduino: Hey RPI, I received a digit and changed it using the offset given: ", number)
    print
```

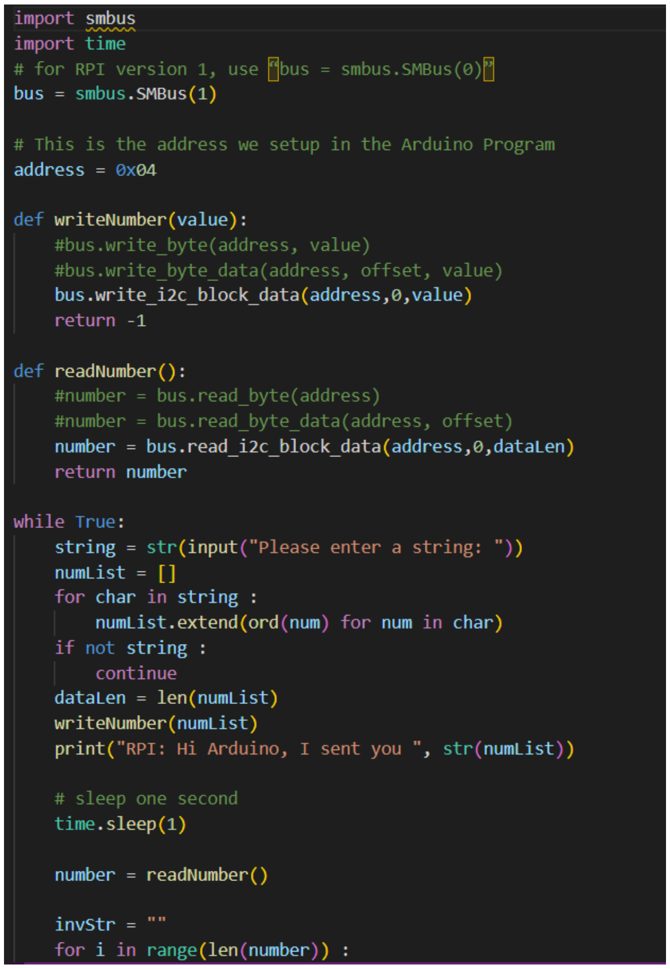
```
#include <Wire.h>

#define SLAVE_ADDRESS 0x04
int number = 0;
int number1 = 0 ;
int state = 0;
int arrayLen = 0 ;
byte data[32] = {0} ;
byte storeData[32] = {0} ;


void setup() {
  pinMode(13, OUTPUT);
  Serial.begin(9600); // start serial for output
  // initialize i2c as slave
  Wire.begin(SLAVE_ADDRESS);

  // define callbacks for i2c communication
  Wire.onReceive(receiveData);
  Wire.onRequest(sendData);

  Serial.println("Ready!");
}

void loop() {
  delay(100);
}

// callback for received data
```

```
void receiveData(int byteCount){
  int i = 0 ;
  while(Wire.available()) {
    data[i] = Wire.read();
    Serial.print(data[i]) ;
    Serial.print(" ") ;
    i++ ;
    arrayLen++ ;

  }
  i-- ;
  Serial.println(" ") ;
  int j = 0 ;
  for (int k = i - 1; k != -1; k--) {
    storeData[j] = data[k+1] ;
    j++ ;
  }
}

// callback for sending data
void sendData(){
  Wire.write(storeData, 32) ;
}
```

<u>Exercise 4</u>


KEYSIGHT TECHNOLOGIES
MSO-X 3012T, MY55280072, 07.50.2021102830: Thu Sep 22 02:16:31 2022
1 2.00V/ 2 2.00V/ 50.00us/ 216.0us Stop 2 1.25V
USB device installed as "usb".
Meas
Freq(1): 8.928kHz
Pk-Pk(1): 4.6V
16.0u 116u 216u 316u 416us
-2.02500V DC 10.0:1
4.70000V DC 10.0:1
2:15 AM Sep 22, 2022

Exercise 5:

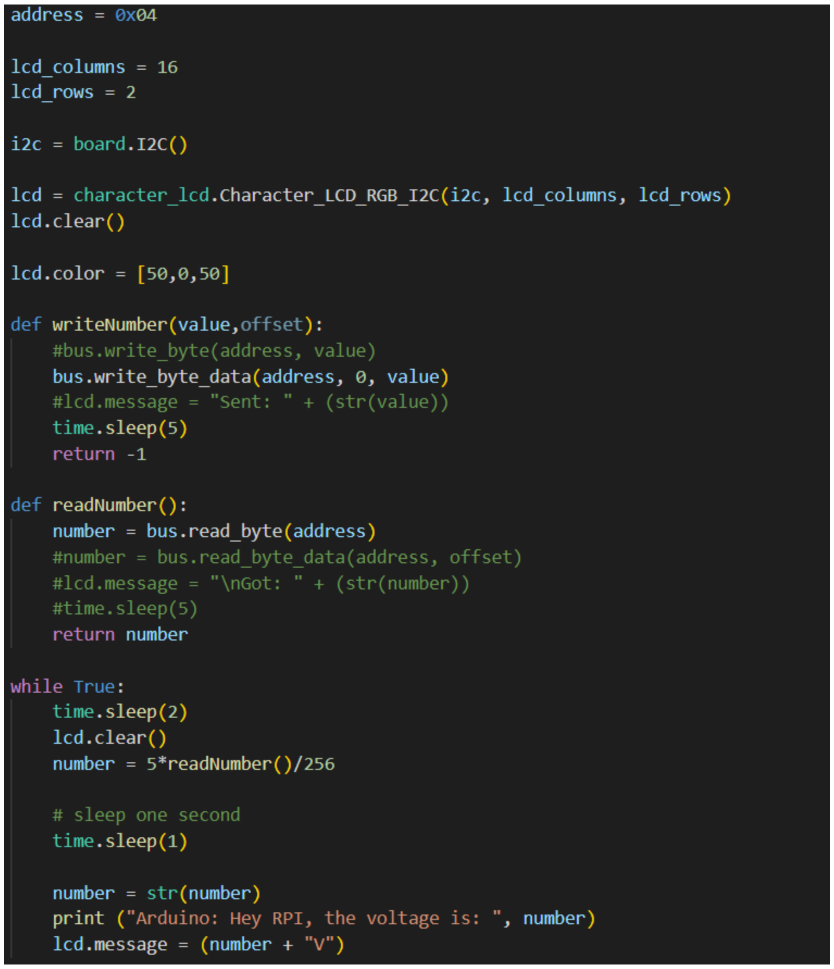


```python
import smbus
import time
# for RPI version 1, use "bus = smbus.SMBus(0)"
bus = smbus.SMBus(1)

# This is the address we setup in the Arduino Program
address = 0x04

def writeNumber(value):
    #bus.write_byte(address, value)
    #bus.write_byte_data(address, offset, value)
    bus.write_i2c_block_data(address,0,value)
    return -1

def readNumber():
    #number = bus.read_byte(address)
    #number = bus.read_byte_data(address, offset)
    number = bus.read_i2c_block_data(address,0,dataLen)
    return number

while True:
    string = str(input("Please enter a string: "))
    numList = []
    for char in string :
        numList.extend(ord(num) for num in char)
    if not string :
        continue
    dataLen = len(numList)
    writeNumber(numList)
    print("RPI: Hi Arduino, I sent you ", str(numList))

    # sleep one second
    time.sleep(1)

    number = readNumber()

    invStr = ""
    for i in range(len(number)) :
```

```
#include <Wire.h>

#define SLAVE_ADDRESS 0x04
int number = 0;
int number1 = 0 ;
int state = 0;
int arrayLen = 0 ;
byte data[32] = {0} ;
byte storeData[32] = {0} ;


void setup() {
  pinMode(13, OUTPUT);
  Serial.begin(9600); // start serial for output
  // initialize i2c as slave
  Wire.begin(SLAVE_ADDRESS);

  // define callbacks for i2c communication
  Wire.onReceive(receiveData);
  Wire.onRequest(sendData);

  Serial.println("Ready!");
}

void loop() {
  delay(100);
}
```

```
// callback for received data
void receiveData(int byteCount){
  int i = 0 ;
  while(Wire.available()) {
    data[i] = Wire.read();
    Serial.print(data[i]) ;
    Serial.print(" ") ;
    i++ ;
    arrayLen++ ;

  }
  i-- ;
  Serial.println(" ") ;
  int j = 0 ;
  for (int k = i - 1; k != -1; k--) {
    storeData[j] = data[k+1] ;
    j++ ;
  }
}

// callback for sending data
void sendData(){
  Wire.write(storeData, 32) ;
}
```

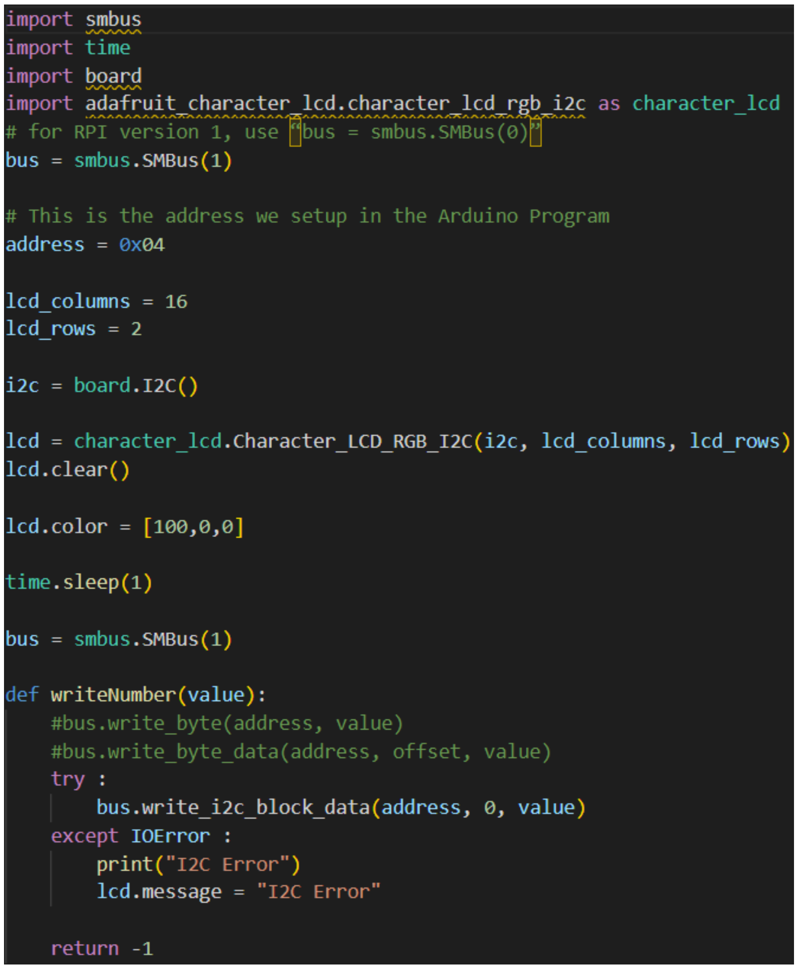
Exercise 6:



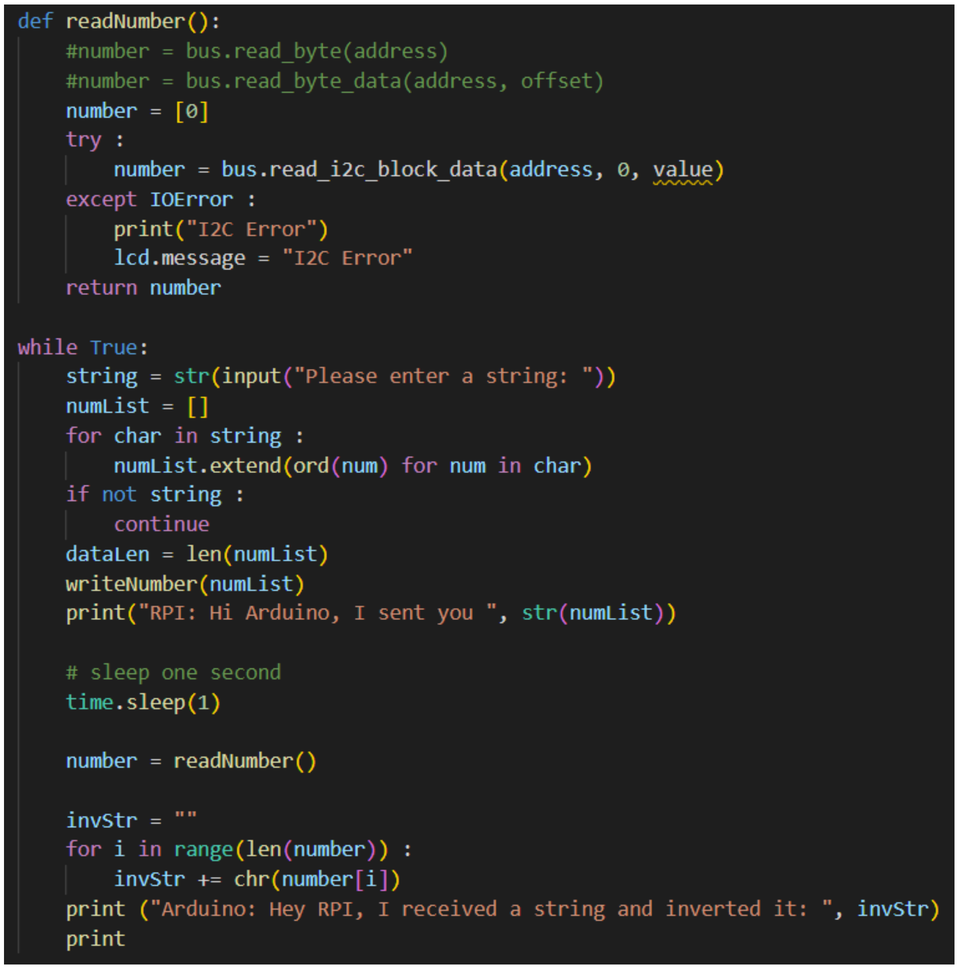
```python
address = 0x04

lcd_columns = 16
lcd_rows = 2

i2c = board.I2C()

lcd = character_lcd.Character_LCD_RGB_I2C(i2c, lcd_columns, lcd_rows)
lcd.clear()

lcd.color = [50,0,50]

def writeNumber(value,offset):
    #bus.write_byte(address, value)
    bus.write_byte_data(address, 0, value)
    #lcd.message = "Sent: " + (str(value))
    time.sleep(5)
    return -1

def readNumber():
    number = bus.read_byte(address)
    #number = bus.read_byte_data(address, offset)
    #lcd.message = "\nGot: " + (str(number))
    #time.sleep(5)
    return number

while True:
    time.sleep(2)
    lcd.clear()
    number = 5*readNumber()/256

    # sleep one second
    time.sleep(1)

    number = str(number)
    print ("Arduino: Hey RPI, the voltage is: ", number)
    lcd.message = (number + "V")
```

```
#include <Wire.h>

#define SLAVE_ADDRESS 0x04
int sensorPin = A0;
int state = 0;
int sensorValue = 0 ;
byte data = 0 ;
void setup() {
  pinMode(13, OUTPUT);
  Serial.begin(9600); // start serial for output
  // initialize i2c as slave
  Wire.begin(SLAVE_ADDRESS);

  // define callbacks for i2c communication
  Wire.onRequest(sendData);
  pinMode(13, OUTPUT) ;
  Serial.println("Ready!");
}

void loop() {
  sensorValue = analogRead(sensorPin) ;
  data = byte(sensorValue / 4) ;
  Wire.write(12) ;
  digitalWrite(13,HIGH) ;
  delay(sensorValue) ;
  Serial.print(data) ;
}
```

Exercise 7:



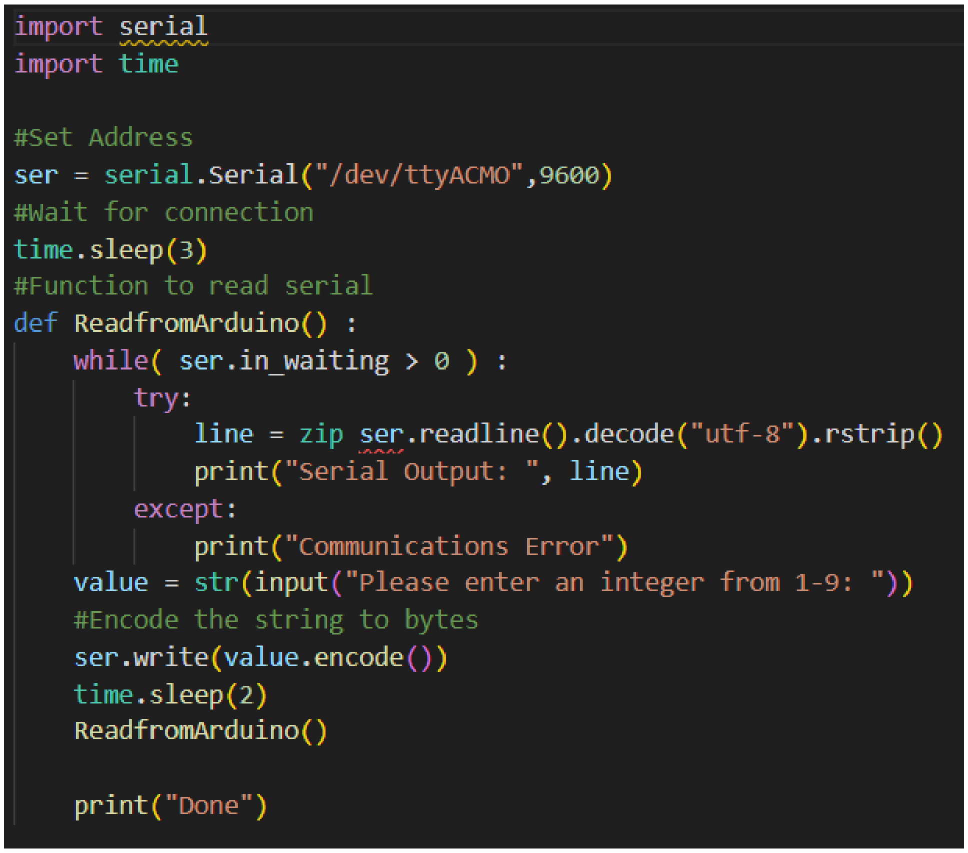
```
import smbus
import time
import board
import adafruit_character_lcd.character_lcd_rgb_i2c as character_lcd
# for RPI version 1, use "bus = smbus.SMBus(0)"
bus = smbus.SMBus(1)

# This is the address we setup in the Arduino Program
address = 0x04

lcd_columns = 16
lcd_rows = 2

i2c = board.I2C()

lcd = character_lcd.Character_LCD_RGB_I2C(i2c, lcd_columns, lcd_rows)
lcd.clear()

lcd.color = [100,0,0]

time.sleep(1)

bus = smbus.SMBus(1)

def writeNumber(value):
    #bus.write_byte(address, value)
    #bus.write_byte_data(address, offset, value)
    try :
        bus.write_i2c_block_data(address, 0, value)
    except IOError :
        print("I2C Error")
        lcd.message = "I2C Error"

    return -1
```



```
def readNumber():
    #number = bus.read_byte(address)
    #number = bus.read_byte_data(address, offset)
    number = [0]
    try :
        number = bus.read_i2c_block_data(address, 0, value)
    except IOError :
        print("I2C Error")
        lcd.message = "I2C Error"
    return number

while True:
    string = str(input("Please enter a string: "))
    numList = []
    for char in string :
        numList.extend(ord(num) for num in char)
    if not string :
        continue
    dataLen = len(numList)
    writeNumber(numList)
    print("RPI: Hi Arduino, I sent you ", str(numList))

    # sleep one second
    time.sleep(1)

    number = readNumber()

    invStr = ""
    for i in range(len(number)) :
        invStr += chr(number[i])
    print ("Arduino: Hey RPI, I received a string and inverted it: ", invStr)
    print
```

```
#include <Wire.h>

#define SLAVE_ADDRESS 0x04
int number = 0;
int number1 = 0 ;
int state = 0;
int arrayLen = 0 ;
byte data[32] = {0} ;
byte storeData[32] = {0} ;


void setup() {
  pinMode(13, OUTPUT);
  Serial.begin(9600); // start serial for output
  // initialize i2c as slave
  Wire.begin(SLAVE_ADDRESS);

  // define callbacks for i2c communication
  Wire.onReceive(receiveData);
  Wire.onRequest(sendData);

  Serial.println("Ready!");
}

void loop() {
  delay(100);
}

// callback for received data
void receiveData(int byteCount){
  int i = 0 ;
  while(Wire.available()) {
    data[i] = Wire.read();
    Serial.print(data[i]) ;
    Serial.print(" ") ;
    i++ ;
    arrayLen++ ;

  }
  i-- ;
  Serial.println(" ") ;
  int j = 0 ;
  for (int k = i - 1; k != -1; k--) {
    storeData[j] = data[k+1] ;
    j++ ;
  }
}

// callback for sending data
void sendData(){
  Wire.write(storeData, 32) ;
}
```

Exercise 8:



```python
import serial
import time

#Set Address
ser = serial.Serial("/dev/ttyACMO",9600)
#Wait for connection
time.sleep(3)
#Function to read serial
def ReadfromArduino() :
    while( ser.in_waiting > 0 ) :
        try:
            line = zip ser.readline().decode("utf-8").rstrip()
            print("Serial Output: ", line)
        except:
            print("Communications Error")
    value = str(input("Please enter an integer from 1-9: "))
    #Encode the string to bytes
    ser.write(value.encode())
    time.sleep(2)
    ReadfromArduino()

    print("Done")
```

```
String data ;
bool dataRead ;

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

void loop() {
  if (dataRead) {
    data = String(data.toInt() + 5) ;
    Serial.print("You sent: ") ;
    Serial.print(data) ;
    dataRead = false ;
  }
}

void serialEvent() {
  if (Serial.available() > 0) {
    data = Serial.readStringUntil("\n") ;
    dataRead = true ;
  }
  Serial.flush() ;
}
```