

Arduino

Lo primero que tenemos que hacer es cargar el archivo en nuestro Arduino para esto debemos tener instalado Arduino ide.

Una vez instalado, se debe abrir el archivo y dar en la opción de cargar archivo.

The image shows a screenshot of the Arduino IDE interface. The title bar at the top reads "filtro_arduino_uno Arduino 1.8.19". Below the title bar is a menu bar with "Archivo", "Editar", "Programa", "Herramientas", and "Ayuda". A toolbar with icons for opening, saving, and uploading files is visible. The main text area contains the following C++ code:

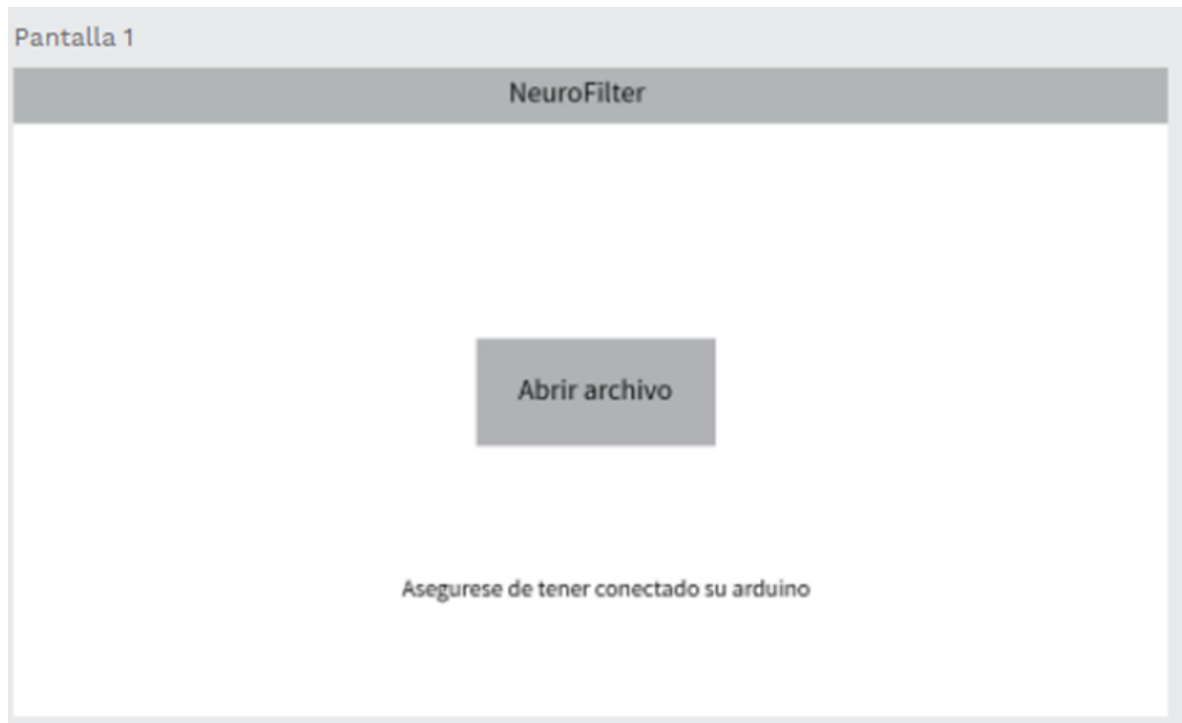
```
1#include <Float64.h>
2void setup() {
3  Serial.begin(9600);
4
5}
6
7void loop() {
8
9  if (Serial.available()) {
10
11    String bytes = Serial.readString();
12    f64 x = atof64(Serial.readString().c_str());
13    x.setDecs(17);
14    Serial.print(F("Float value: "));
15    Serial.println(bytes);
16  }
17}
```

The status bar at the bottom indicates "Arduino Uno en COM3".

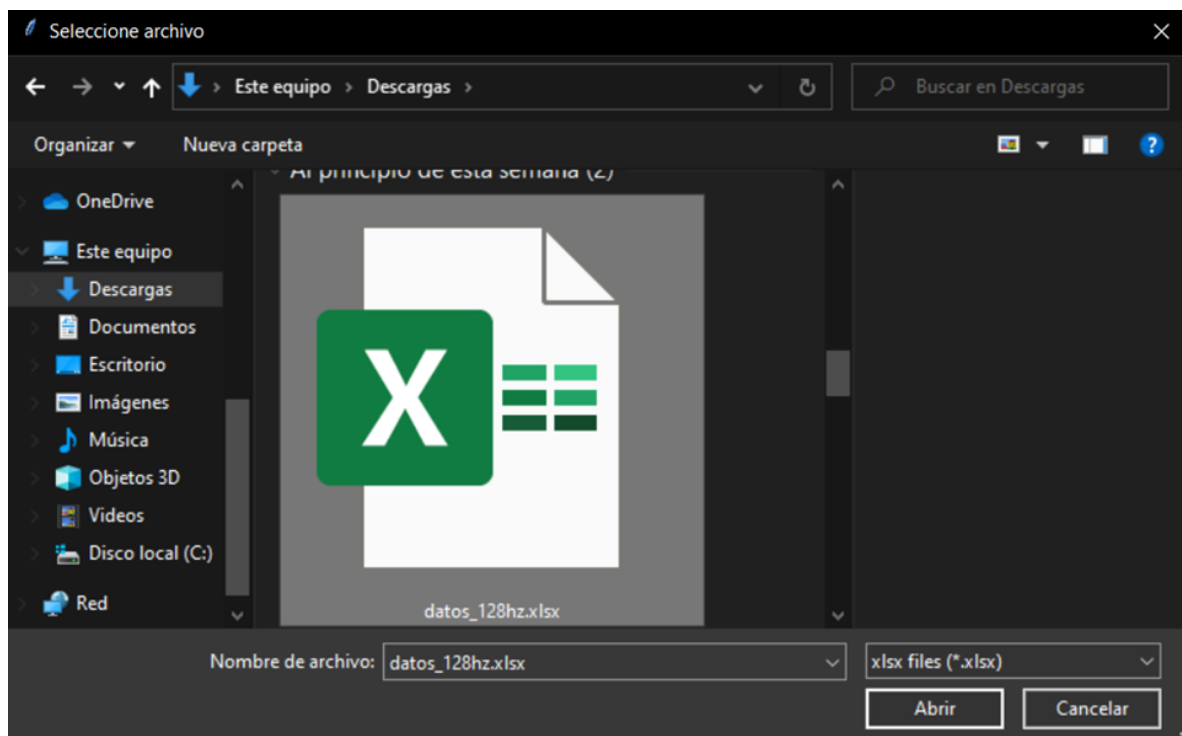
Después podemos cerrar el programa.

NeuralFilter

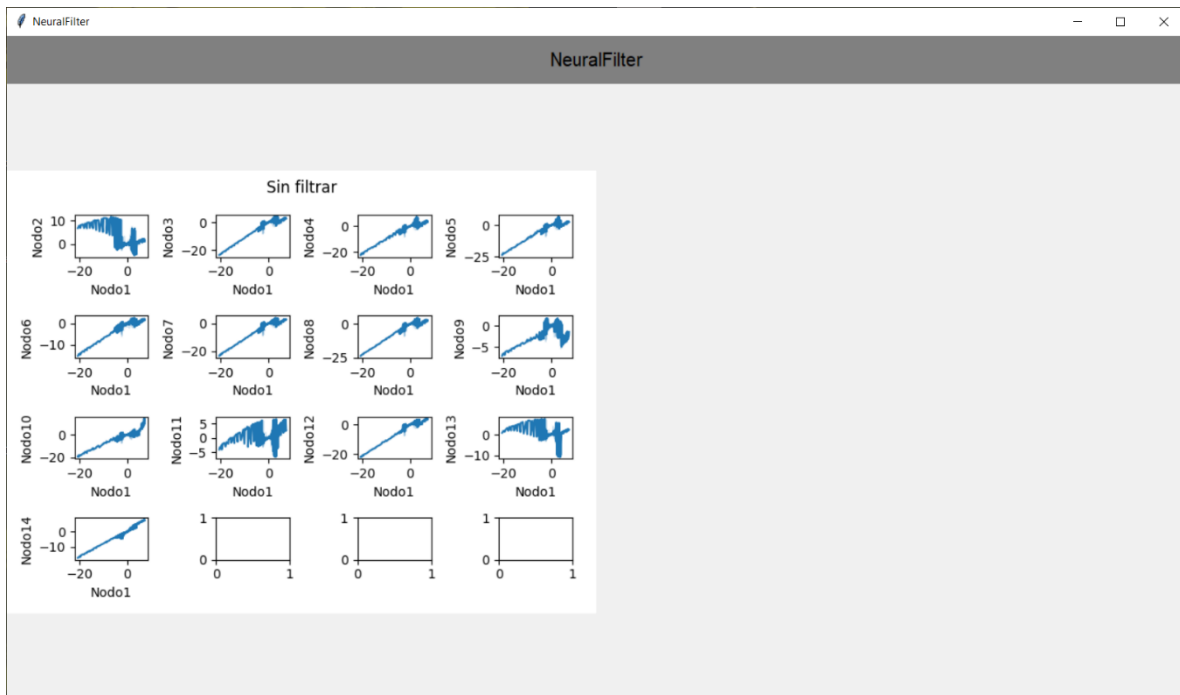
La interfaz de programa constaría de dos pantallas, esta sería la primera pantalla donde se pregunta al usuario sobre el archivo Excel que va filtrar.



El usuario deberá buscar el archivo y el programa enviará los datos a Arduino.



Después Saldrá una ventana con los datos graficados.



Funcionamiento

```
1 import serial.tools.list_ports
2
3 # esta funcion simplemente regresa el puerto que usa el arduino para conectarse
4
5 def get_ports():
6     # esto llama a un objeto con la informacion del arduino
7     ports = serial.tools.list_ports.comports()
8     print('Objeto', ports)
9     # retorna el objeto
10    return ports
11
12 # esta funcion a va regresar el port que utilizar el arduino
13 # esta funcion necesita el objeto
14
15 def conectar_arduino(ports_encontrados):
16     # variable donde se almacena el port
17     comm_port = "None"
18     # en caso de haya mas de un arduino conectado va a tomar ese numero
19     # es el numero de veces que se va a repetir el ciclo
20     num_connections = len(ports_encontrados)
21
22     for i in range(0, num_connections):
```

Esta parte de código se encarga de encontrar el puerto que usa el usuario, pero esta limitado a arduinos originales.

The screenshot shows the PyCharm IDE interface. The main editor displays the file `graficacion.py` with the following Python code:

```
1 import pandas as pd
2 import seaborn as sns
3
4 from matplotlib import pyplot as plt
5 from src.main import archivoRuta
6 from src.envio_datos import *
7 from src.conexion import *
8
9
10 def graficador_seaborn():
11     # variable para colocar la ruta del archivo
12     ruta = ""
13     # variable para conseguir la ruta del archivo
14     datos = pd.read_excel(ruta.join(archivoRuta))
15     nuevosDatos = datos[
16         ['Nodo1', 'Nodo2', 'Nodo3', 'Nodo4', 'Nodo5', 'Nodo6', 'Nodo7', 'Nodo8', 'Nodo9', 'Nodo10', 'Nodo11', 'Nodo12',
17          'Nodo13', 'Nodo14']]
18     fig, axes = plt.subplots(4,4)
19     for ax_i in zip(axes.ravel(), nuevosDatos.columns[1:]):
20         g = sns.lineplot(x='Nodo1', y=_i, ax=ax, data=nuevosDatos)
21
22     fig.suptitle('Sin filtrar')
23     fig.tight_layout()
24     return fig
25
26 graficador_seaborn()
```

The left sidebar shows the project structure with folders like `pythonProject`, `arduino`, `data`, `docs`, `src`, and `tests`. The bottom panel shows the Run console output:

```
Run: main () x
COM3
Arduino conectado COM3
Conexion Hecha
Entro al graficador
Traceback (most recent call last):
```

The status bar at the bottom indicates the file encoding is UTF-8, the Python version is 3.10, and the current file is `graficacion.py`.

Esta parte del código se encarga de graficar los datos sin filtrar.