

TRABAJO FINAL - COLOQUIO

INFORMÁTICA

5 DE NOVIEMBRE DE 2019

INGENIERÍA MECATRÓNICA

Universidad Nacional de Cuyo

INTEGRANTES:

Léa Laxague V0687

Maximiliano Reinoso 11754

Lilia García Hiramatsu 10973

Adriano Germán Eula 12336

Andrés Muñoz Pampillon 12850

En el contexto de la finalización de la cátedra Informática, teníamos que realizar un trabajo integrador en el cual se abarcara en contenido total de la misma. Para ello, se utilizó la Raspberry PI, una breadboard armada con un procesador, un sensor de temperatura, y tres LEDs.

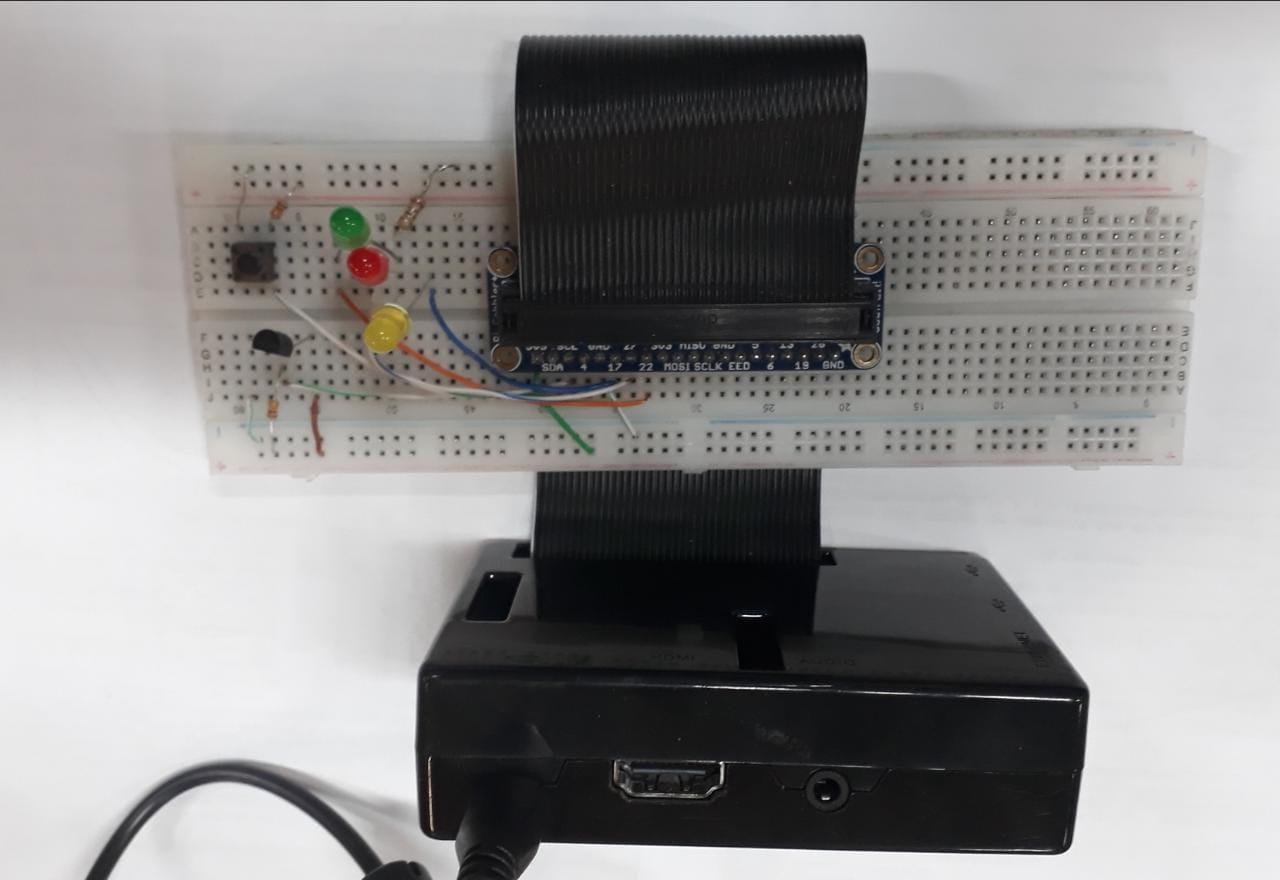
El proyecto consta de tres partes:

* Parte A: realizar el monitoreo y seguimiento de la temperatura mediante el uso de Python.
* Parte B: Analizar los datos obtenidos mediante el lenguaje C.
* Parte C: Realizar una visualización de las lecturas en dos partes: configurar un socket TCP/IP que después envía los datos por el mismo, lo que crea un sistema « servidor / cliente ». Luego hacer un programa en una computadora que se conectará al socket previamente establecido en la Raspberry PI y procederá a la visualización de los datos por allí transferidos.

**La placa Raspberry Pi:**

Esta placa está compuesta por:

* Un procesador 1.2 GHz quad core ARM Cortex A53
* Un GPU VideoCore IV para reproducir hasta 1080p
* 1 GB de RAM
* Salida de video HDMI y audio estéreo
* Cuatro puertos USB 2.0 y un puerto Ethernet
* Wifi y Bluetooth



Además hay tres LEDs que nos permiten conocer el estado del sistema, un sensor que mide temperaturas desde -55°C a 125°C, y un pulsador que nos permite cambiar el tiempo de medida.

**Realización de la parte A:**

Teníamos que realizar una estación de monitoreo y seguimiento de temperatura con los siguientes requisitos:

- Debe sensar temperatura cada un ciclo de X segundos. Por defecto el ciclo es de 3.5 segundos.

* En cada ciclo la estación muestra por consola la temperatura que representa la última temperatura leída y el promedio de los últimos N de los eventos.
* El ciclo se configura mediante un pulsador que cuando es apretado comienza a medir en milisegundos, y luego cuando se suelta se cierra el conteo. Se le informa al usuario de que el valor ha sido alterado. El ciclo mínimo es de 2.5 segundos y el ciclo máximo es de 10 segundos. Una vez concluido el cálculo del ciclo se informa por pantalla el nuevo tiempo del ciclo.
* La estación además posee un semáforo con 3 leds los cuales indican la tendencia de cambio de la temperatura, es decir :
* Un led rojo enciende cuando el valor de la lectura en un ciclo supera en un X% del promedio de los últimos N eventos (tendencia en alta).
* Un led amarillo indica que la temperatura se mantiene estable, es decir se encuentra dentro del promedio (+-X%)
* Un led verde enciende cuando la temperatura disminuye en un X% del promedio de los últimos N eventos (tendencia en baja).
* Los 3 leds de tendencia se mantendrán prendidos cuando no se tenga la suficiente información para analizar la tendencia. Esto es así cuando no se dispongan de al menos el 5% de los eventos necesarios para calcular el promedio.
* Una última tarea que realiza esta estación de monitoreo es ir registrando en un archivo por cada ciclo:
* Lectura de temperatura
* Tendencia detectada (para cada uno de los 4 estados posibles)
* Fecha y Hora en que se realizó la lectura

Entradas:

* Dispositivos de entrada: pulsador y sensor de temperatura.
* Cantidad de eventos mínimos (Dato ingresado por usuario)
* Variación porcentual (Dato ingresado por usuario)
* Lectura de fecha y hora (Entrada desde el sistema)

Salidas:

* Dispositivos de salida: LEDs (led rojo, led amarillo, led verde)
* Aviso de cambio de tiempo de ciclo (Central de Monitoreo Apagado, Nuevo tiempo de ciclo, Error)
* Ultima temperatura tomada
* Promedio de las N temperaturas tomadas
* Salida de datos a un archivo

Información relevante:

* El ciclo inicia en 3,5 segundos por defecto
* Tiempo de ciclo (mínimo, máximo)
* Al mantener presionado el pulsador se define un nuevo tiempo de muestreo en mili segundos
* El mismo pulsador apaga el monitoreo si es presionado un tiempo menor al tiempo mínimo
* El led rojo indica un aumento de temperatura un X% con respecto a la media
* El led amarillo indica temperatura estable y esta se encuentra entre -X% y +X% respecto a la media
* El led verde indica un descenso de temperatura un -X% con respecto a la media
* Cuando un led se enciende los demás se apagan
* Se mantienen los 3 led encendidos cuando la cantidad de muestras es menor a la mínima
* Porcentaje del promedio que indique las tendencias
* Almacenar datos en un archivo .txt (temperatura, tendencia, fecha y hora)

Requerimientos Software:

* Solicitar datos al usuario por terminal
* Sentencias para mostrar información
* Sistema que mida la temperatura
* Entorno de desarrollo
* Terminal

Requerimientos Hardware:

* Tres leds de distinto color
* Resistencias
* Sensor de temperatura
* Pulsador
* Memoria para almacenar los datos
* Conexión serial o medio para mostrar los datos por terminal
* Conexión para los leds
* Conexión para transmitir los datos

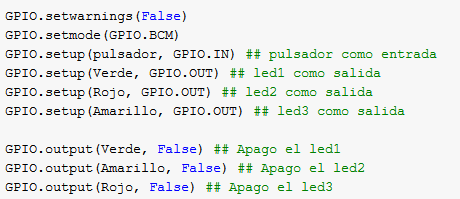
Restricciones:

* Restricciones de temperatura del sensor, capacidad y calidad. (Rango de temperatura desde -55°C hasta 125°C con precisión de +-2°C).
* Mientras se toma lecturas de temperatura no se puede encender más de 1 led a la vez después de haber superado la muestra mínima.
* El programa finaliza si el pulsador se mantiene apretado durante un tiempo menor al del ciclo mínimo (2500 milisegundos)
* Tiempo máximo que puede adoptar el ciclo (10000 milisegundos)

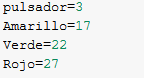
Pseudocódigo:

1. Inicio
2. Définir GPIO de la placa





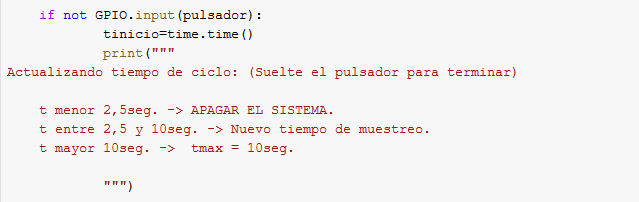
1. Définir pines



1. Cargar constantes



1. Solicitar datos al usuario
2. Définir variables
3. Définir variable de estado
4. Ejecutar las variables de estado en un bucle:
   1. Pulsador
      1. Si pulsador presionado tomar tiempo
      2. Si pulsador se suelta tomar otro tiempo
      3. Hacer diferencia de los tiempos tomados
      4. Si diferencia de tiempos es menor al tiempo mínimo: se apaga el equipo, los LEDs se apagan de manera secuencial.
      5. Si diferencia de tiempos está entre el mínimo y el máximo
         1. Definir un nuevo tiempo de ciclo
         2. Decir “Nuevo tiempo de muestreo”
      6. Si diferencia de tiempos es mayor al máximo: se mantiene con el máximo valor de 10 segundos.



* 1. Si se cumplió el tiempo de ciclo (Comparando con el tiempo real y el tiempo de la última medida)
     1. Llamar a la función que determina la lectura del termistor.
     2. Leer y mostrar temperatura
     3. Tomar datos de tiempo en milisegundos
     4. Tomar fecha y hora de la medición
     5. Escribir temperatura, fecha y hora en archivo de datos
     6. Calcular promedio
        1. Comparar temperatura con promedio histórico
           1. Si la temperatura es mayor a un X% del promedio

Encender led rojo

Escribir en el archivo “Tendencia Alta”

* + - * 1. Si la temperatura es menor a un – X% del promedio

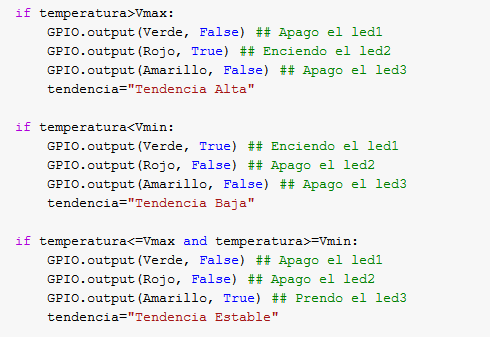
Encender led verde

Escribir en el archivo “Tendencia Baja”

* + - * 1. Si la temperatura está entre -X% y un X% del promedio

Encender led amarillo

Escribir en el archivo “Tendencia Estable”



1. Limpiar GPIO



*Comentario: Sí se presiona el pulsador un tiempo suficientemente corto de modo tal que el código no llega a tomar y entonces no se va a dar cuenta que el pulsador fue pulsado.*

**Realización de la parte B:**

En esta parte tenemos que tomar los datos del archivo creado en la parte A y los analizamos estadísticamente. Guardamos los datos en una estructura dinámica, es este caso utilizamos una lista. Dicha lista contiene las lecturas de temperaturas extraídas del archivo. Utilizamos una lista ya que nos pareció la estructura de datos más eficiente para guardar nuestros datos, y así, poder analizarlos. Esto es porque, al tratarse de una estructura de datos dinámica, nos permite modificar y/o agregar valores en tiempo de ejecución. Además, para este caso, es más eficiente que otras estructuras dinámicas, como las pilas y las colas, porque sólo necesitamos agregar los datos nuevos al final de la lista y además, recorrerla para analizar dichos datos. Con una pila ó con una cola sería demasiado engorroso llevar a cabo las tareas mencionadas.

En la parte en que abrimos el archivo para extraer el dato de la temperatura y la fecha correspondientes, tuvimos que crear arreglos del tipo string para poder extraer sólo los caracteres correspondientes a dichos datos, y desechar la parte de la línea en que se muestra el promedio de los datos y la tendencia, ya que esos últimos no son relevantes para el análisis. En el caso del string que contiene al valor de la temperatura, debimos hacer una conversión a flotanto mediante la función atof. Finalmente este es el valor que se guarda en el nodo de la lista.

A continuación, realizamos el análisis estadístico correspondiente a esta parte del proyecto. Realizamos los cálculos de la media, la mediana, la desviación estándar y la moda. Además, debíamos encontrar la menor y la mayor temperatura leídas por el sensor e imprimir por consola dichos valores junto a la fecha y hora a la que se sensaron.

Para el cálculo de la mediana, debíamos ordenar los datos de la lista de menor a mayor para realizar su cálculo. Esto fue un problema, pero lo resolvimos con el algoritmo de ordenamiento : el método burbuja. Este no es el más eficiente de todos los métodos pero funcionó perfectamente para la tarea que debíamos realizar. Este método consiste en tomar un nodo y el siguiente de la lista, por empezar, el primero y el siguiente a él, y los compara. El que resulta menor, se guarda en un nodo auxiliar y luego se actualizan los valores. Esto se realiza en un bucle, con el fin de recorrer toda la lista y así, poder comparar todos los valores. Al final, se obtiene la lista ordenada, del menor valor al mayor valor. Una vez hecho esto, pudimos realizar el cálculo de la mediana facilmente.

Una función que nos fue muy útil fue la de « obtener\_indice » porque al encontrar la posición del menor valor y del mayor valor de temperatura, pudimos guardar dichos índices de las posiciones respectivas para luego buscar en el archivo la fecha de las posiciones correspondientes.

Utilizamos subrutinas en el código para obtener mayor legibilidad del mismo y hacerlo más eficiente.

Finalmente, imprimimos por pantalla todos los datos pedidos. De esta manera pudimos obtener una mejor información de los datos que se midieron por el sensor.

**Realización de la parte C:**

En esta parte debemos desarrollar un código en Python capaz de establecer una conexión mediante un protocolo TCP-IP por el cual se enviará último valor sensado hacia otro dispositivo, el cual se debe graficar en tiempo real los mismos pudiendo así representar a lo largo de la ejecución del programa todos los valores que se reciban.

Pre-análisis:

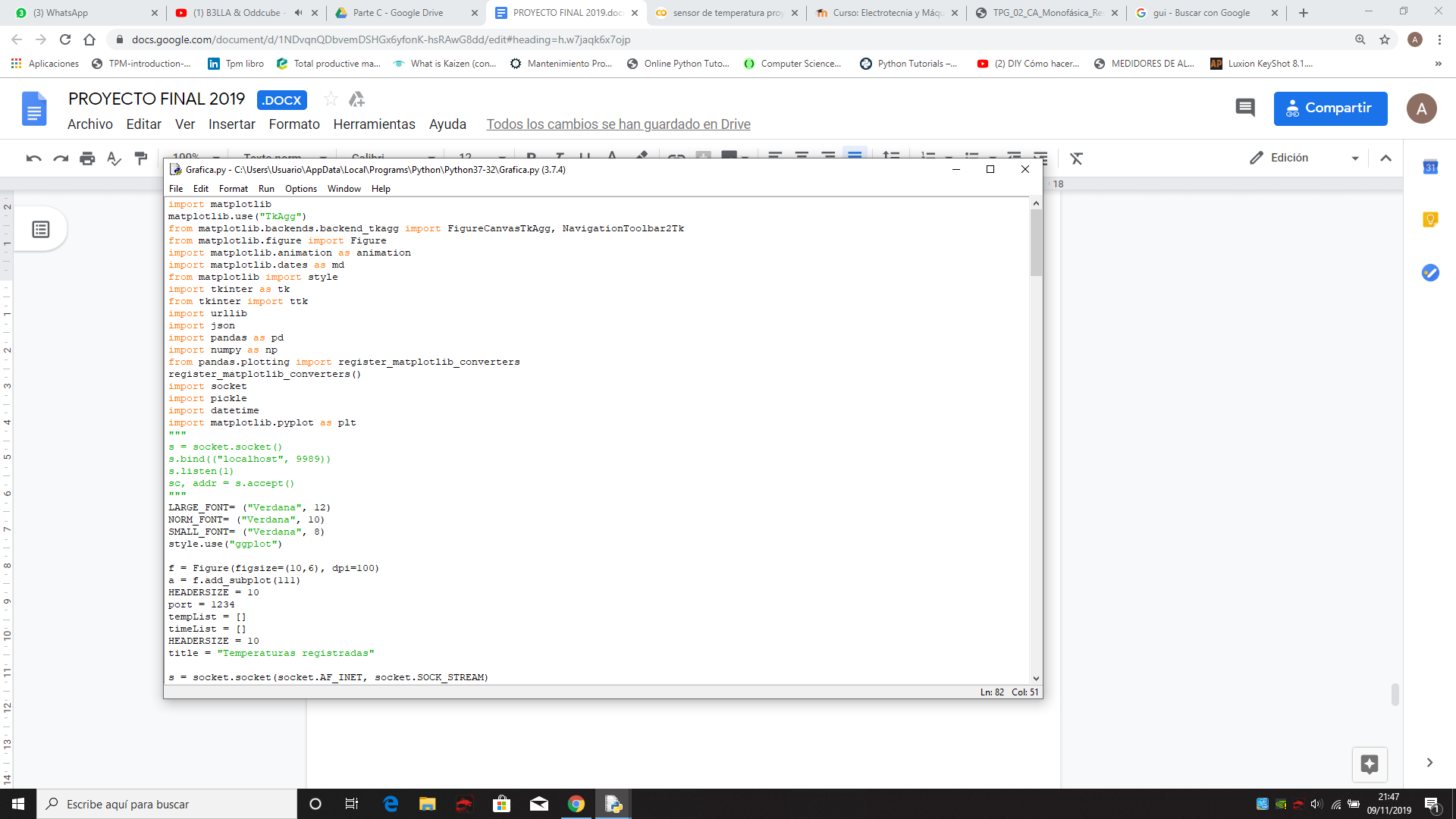
Antes de comenzar la programación del código se evaluó cómo estructurar el mismo de manera óptima por lo que se decide implementar una GUI (graphical user interface). Orientando así el uso del programa a personas que no necesitan necesariamente habilidades de programación o requieran de modificar el código fuente para realizar pequeños cambios.

Enmarcando nuestro programa bajo la estructura de una GUI se dispone de mayores herramientas y recursos para realizar gráficos, y agregar funciones que faciliten la visualización del mismo como así opciones extra que mejoran la experiencia del usuario.

Libreiras:

La ejecución de este código se basa principalmente en 4 librerías de python las cuales nos permiten desarrollar nuestro programa.

* tkinter: Esta librería nos permite desarrollar nuestra GUI, brindanonos funciones y comandos para asignar botones, cuadros, barras y un lienzo en el cual poder desplegar gráficos.
* socket: Usando esta librería podemos usar los puertos y conectarnos por TCP-IP al servidor.
* pickle: El módulo pickle implementa protocolos binarios para serializar y deserializar una estructura de objeto Python.
* matplotlib: Matplotlib es una biblioteca de trazado 2D de Python. Esta misma nos permite graficar los valores de la temperatura con respecto al tiempo. Permitiendo cambiar valores tales como el título, color, tipo de trazado, nombre de los ejes y sus etiquetas entre otros.



Estructura:

El programa se compone de una clase principal la cual es nuestro

Aquí está el código realizado en las partes A, B y C:

[https://colab.research.google.com/drive/1givVUL6WBshjr5M-ccWQTQ9GEtC3rl7U#scrollTo=ezpTZgy7hRfN](https://colab.research.google.com/drive/1givVUL6WBshjr5M-ccWQTQ9GEtC3rl7U" \l "scrollTo=ezpTZgy7hRfN)