**5. óra: Szerelem mérő**

1. 5 perces beszámoló tetszőleges tartalommal és üzenettel valamint tetszőlegesen választott retorikai eszközökkel. Nekünk kell eldönteni, hogy mi lehetett az üzenet. Figyelni kell a beszéd felépítésére (bevezető, fő rész és konklúzió) és arra, hogy a tartalom minden eleme támogassa az üzenetet, illetve, hogy használjunk retorikai eszközöket.
2. 3 perces véleményalkotás  
   - bevezető, fő rész, konklúzió  
   - 2 pozitív és egy konstruktív javaslat  
   - konklúzióban összefoglalni mi lehetett az üzenet, volt-e retorikai eszköz használva és adni még egy pozitív megjegyzést.
3. Analóg hőmérséklet mérése raspberryvel:
   1. *Bevezető*:  
      Most már rendelkezünk egy szerelem mérő eszközzel, ami furfangos módon egy termisztor ellenállásának hőre való változásából a rajta eső feszültséget méri. Ezt még leírni is körülményes. Sőt beállítani a megfelelő feszültségparamétereket is az. Ezért jobb lenne, ha igazából a mért feszültséget át tudnánk számolni hőmérsékletté. Evvel a módszerrel az alanyunktól származó hőt mérnénk valamilyen formában. Mivel tudjuk, hogy az emberi test hőmérséklete ideális esetben 36 Celsius fok körül van, így könnyebb lesz feltételek szabni a LED-ek vezérlésére.
   2. *Előkészületek*:  
      Készítsünk ki próbapaneleket, jumper kábeleket, 10 kOhm ellenállásokat a thermisztornak, NTCC 10k thermisztorokat, piros, sárga és zöld LEDeket, hozzájuk tartozó ellenállásokat kb. 200 Ohm és az MCP3008 ADCt. Emellé szükség lesz mérőműszerre is.   
      - Rakjuk össze az áramkört. Teszteljük le műszerrel, hogy kb. 1.6 Vot mérünk-e a thermisztor lábain (feltéve, hogy szobahőmérséklet van, kb. 25 fok). Mindenesetre, amit mérünk a lábakon, azt kell majd az MCP3008al is mérnünk.   
      - Teszteljük le a műszerrel, hogy működnek-e illetve pythonból is, hogy jó e az áramkör.  
      - Teszteljük le az MCP3008at is pythonból, hogy azt olvas-e amit a műszer.   
        
      *Részletekre bontás:*Elkészíteni az áramkört.
      * Importálni az MCP3008, LED objektumokat, numpy csomagot, függvényt az interpolálásra és fájl beolvasó függvényt.
      * Inicializálni a LED-eket és az ADC-t.
      * Definiálni egy függvényt, ami egy megadott fájlból kinyeri a thermisztor hőmérséklet-ellenállás függését.
      * Definiálunk egy függvényt ami a mért feszültséget átalakítja ellenállássá a feszültségosztó elve alapján.
      * Definiálunk egy függvényt, hogy a hőmérséklet-ellenállás függés ismeretlen pontjait interpoláljuk.
      * Definiálunk egy függvényt a LED-ek vezérlésére.
      * Beolvassuk a hőmérséklet-ellenállás függését, a fent definiált függvénnyel.
      * Egy végtelen while ciklusban mérjük a feszültséget, kiszámoljuk az ellenállás változást, abból az aktuális hőmérsékletet, és a hőmérséklet alapján pedig vezéreljük a LED-eket.
   3. *Elektronika:*  
      - Készítsük el az áramkört. Érdemes kihasználni a közös földelést és tápot, hiszen sok dróttal dolgozunk majd.  
      - Vezessük le, hogyan számoljuk ki az ellenállást a mért feszültségből.
   4. *Kód:*  
      - Egeszítsük ki a lenti kódvázlatot.  
      - Vegyük át a string műveleteket különös tekintettel a *strip* és *split* metódusokra.  
      - Vegyük át a lista műveleteket megemlítve a *pop* és az append metódust is.   
      - Vegyük át az interpolálást és nézzünk rá példákat:  
        
      f = interp1d(x, y)

new\_x = np.array([-1.5, 3.5])

f(new\_x)  
  
- Nézzük át a műveleteket a *numpy array*el:  
  
a = np.array(a) # numpy.array letrehozasa

print(a)

print(type(a))

print(a\*2) # szamsor elemeinek duplazasa

print(a\*\*2) # szamsor elemeinek negyzetre emelese

print(np.sqrt(a)) # szamsor elemeinek negyzetgyoke  
  
- Törekedjünk arra, hogy ha van rá mód akkor *for* ciklusban kezeljük a LEDek működését.

* 1. *Tesztelés:*  
     - teszteljük a thermisztorokat a műszerrel, illetve teszteljük a programot.
  2. *Fejlesztés:*  
     - Írjuk át egy fájlba azt a függvényt amivel kinyerjük a hőmérséklet-ellenállás összefüggést és importáljuk be azt a függvényt is.

- Írjuk át a lightning függvényt úgy, hogy bemenő paraméterként megadhassuk a LED-eket vezérlő hőmérséklet limitek értékeit.

Kód minta:

# importald az MCP3008 es LED klasszokat a gpiozero csomagbol

import numpy as np

from scipy.interpolate import interp1d

# importald a read\_temp\_raw fuggvenyt a temperature\_functions fajlbol

mcp = # inicializald az ADC konvertert a megfelelo csatornaval

red = # inicializald a piros ledet

yellow = # inicializald a sarga ledet

green = # inicializald a zold ledet

def read\_mapping(name):

    data = # olvasd be a name fajl tartalmat a read\_temp\_raw

    # a data listanak az elso eleme a fejlecet tartalmazza, igy ki kell venni a listabol a pop metodussal

    tc = # inicializalj egy ures listat a homersekleti ertekek reszere

    r = # inicializalj egy ures listat az ellenallasi ertekek reszere

    for line in data:

        clean\_data = # minden sor ket szamot tartalmaz ami vesszovel van elvalasztva. hasznald a strip metodust az ujsorok es az ures helyek eltuntetesere, majd a split metodust, hogy a stringet szetvagd a vesszok menten

        # EZT A SORT NE SZERKESZD: a clean\_data ket elemet fog tartalmazni, az elso a homerseklet, a masodik az ellenallas kOhmban kifejezve, string formatumban mind a ketto

        # bovitsd a tc listat a clean\_data elso elemevel (homerseklet), de vigyazz, hogy a float paranccsal at kell alakitani a stringet szamma

        # bovitsd az r listat a clean\_data masodik elemevel (ellenallas), de vigyazz, hogy a float paranccsal at kell alakitani a stringet szamma, valamint 1000el szorozni kell, hogy Ohmba kapjuk meg az erteket

    return np.array(tc), np.array(r)

def voltage2resistance(v\_out, v\_in=3.3, r=9.82e3):

    return r \* v\_out / (v\_in - v\_out)

def interpolate\_temperature(resistance, tc, r):

    f = interp1d(r,tc)

    return f(resistance)

def lightning(temp, leds=[green, yellow, red]):

    if ......:  # ha homerseklet (temp) kisebb mint 23 fok:

        # az osszes led kikapcsolt allapotban marad

    elif ....... : # ha temp < 28 es temp >= 23, akkor a zold led vilagit, a tobbi nem

        #

        #

        #

    elif ...... : # ha temp < 32 es temp >= 28 akkor csak a piros nem vilagit

        #

        #

        #

    elif ........ : # ha a homerseklet nagyobb vagy egyenlo mint 32 fok akkor mindegyik vilagit

        # mindegyik led vilagit

tc, resistance = # a read\_mapping fuggvennyel olvasd be az ntcc.csv nevu fajl tartalmat

while True:

    rt = # a voltage2resistance fuggvennyel alakitsd at a mert feszultseget (mcp.voltage) ellenallasa. A v\_in erteke legyen 3.3 V, az r erteke pedig 9820 Ohm)

    t\_therm = # becsuld meg az aktualis ellenellasertekhez tartozo homersekletet az interpolate\_temperature fuggvennyel

    # inditsd el a lightning fuggvenyt a kiszamolt homerseklettel, figyelj, a ledek sorrendjere a listaban, zold az elso, aztan sarga es vegul piros

    print(t\_therm)