**4. óra: Hőmérséklet plottolása**

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. Hőmérséklet kimentése fájlba:
   1. *Bevezető*:  
      Sikerült elkészítened a szüleidnek a programot ami elmenti egy fájlba az egy időszakban mért hőmérsékleti eredményeket. A szüleid egyszerre örültek is és méltatlankodtak az eredményen. Örültek, mert büszkék voltak rád, hogy meg tudtad csinálni, de méltatlankodtak, mert a vesszővel elválasztott számok számukra kicsit zagyvaságnak tűnt. Nehezen látták át. Mivel vizuális típusok a szüleid, megkértek, hogy ábrázold egy grafikonon a fájl tartalmát, azaz a hőmérsékletet Celsius fokban az idő függvényében.  
      Miután látták, hogy lehet ábrázolni, kiötlötték, hogy jó lenne, ha folyamatosan látnánk az aktuális és az elmúlt néhány hőmérsékletet egy ábrán frissülni.
   2. *Előkészületek*:  
      Készítsünk ki próbapaneleket, jumper kábeleket, 4.7kOhm ellenállásokat, DS18B20 thermisztorokat. Emellé szükség lesz mérőműszerre is.   
      - Át kell beszélni a thermisztorok fontosabb tulajdonságait, pl. Mekkora feszültséget (teljesítményt) lehet rá adni.   
      - Ennél a projektnél főként a programozáson van a hangsúly.  
        
      *Részletekre bontás:*Elkészíteni az áramkört.
      * Előkészíteni a kommunikációt a thermisztorral (modprobe parancsok futtatása).
      * Definiálni a hőmérsékletet tartalmazó fájl lokációját.
      * Kiolvasni a fájl tartalmát az *open* parancs segítségével.
      * Definiálni egy függvényt ami a fájlból kinyeri a hőmérsékletet.
      * Definiálni egy függvényt, ami egy listába készíti és frissíti a kimentendő információt (dátum datetime formában, celsius fok float formában)
      * Inicializálni a plottolást.
      * A *while* ciklusban kiolvasni az adatokat majd frissíteni a plottot.
   3. *Elektronika:*  
      - Készítsük el a feszültségelosztó kapcsolást.
   4. *Kód:*  
      - Elevenítsük fel a *datetime* csomag műveleteivel (helyezzünk hangsúlyt az idő string formába alakításra):

*Kód:*  
  
now = dt.datetime.today() *# elmentjuk a mostani pillanatot,*

*# EV-HONAP-NAP ORA:PERC:MS*

print(now)

tomorrow = dt.datetime(2021,2,17) *# letrehozunk egy datetime valtozot*

print(tomorrow)

print(now.minute) *# lekerjuk a valtozo PERC erteket*

print(type(now)) *# kinyomtatjuk a datetime tipusat*

now\_string = now.strftime('**%d**/%B/%Y') *# atalakitjuk a datetime objektumot*

*# stringge NAP/HONAP/EV alakban*

print(now\_string)

print(type(now\_string))

date\_str = '1999-08-11 15:12:33' *# string formaban idopont definialas*

*# string idoforma datetime formava alakitasa*

date\_dt = dt.datetime.strptime(date\_str, '%Y-%m-**%d** %H:%M:%S')

print(date\_dt)

print(date\_dt.month)

print(type(date\_dt))

- fájl műveletek gyakorlása, fájl tartalmának kiolvasása, pl. *f = open(filename, ‘a’)*  
- gyakoroljuk a string műveleteket, lásd a példákat a leírásban.  
- Gyakoroljuk a plottolás:  
- ha bemutattuk a program minden alrészletét, akkor fejezzék be az elkezdett részprogramot.

**import** **matplotlib.pyplot** **as** **plt**

**import** **numpy** **as** **np**

x = np.linspace(-5,5,50) *# 50 elemű array létrehozása -5 és 5 között*

*# y értékek létrehozása x függvényében*

y1 = 2\*x+4

y2 = -2\*x\*\*2 + 3\*x - 1

y3 = 2\*np.cos(x/2)+0.5

*# a plot függvénnyel lehet ábrázolni függvényeket; a label paraméter*

*# értéke határozza meg, hogy mi jelenik meg a jelmagyarázatban*

plt.plot(x,y1, label = '2\*x+4')

*# harmadik paraméterként megadtuk, hogy szaggatott vonalat (--) szeretnénk*

*# diamant markerokkal (d)*

plt.plot(x,y2, '--d', label = '$-2\*x^2 + 3\*x - 1$')

*# harmadik paraméterként megadtuk, hogy négyzet markert (s) szeretnénk*

*# fekete színben (k)*

plt.plot(x,y3, 'sk', label = '$2\cos(x/2)+0.5$')

plt.legend() *# megjeleníti a jelmagyarázatot*

plt.title('Függvények') *# megjeleníti az ábra címét*

plt.xlabel('x') *# elnevezi az x tengelyt*

plt.ylabel('y') *# elnevezi az y tengelyt*

plt.show() *# megjeleníti az ábrát*

* 1. *Tesztelés:*  
     - teszteljük a thermisztorokat a műszerrel, illetve teszteljük a programot.
  2. *Fejlesztés:*  
     - Változtassuk meg, hogy hány pontot ábrázoljunk egyszerre.

- Jelenítsük meg a jelmagyarázatot is a grafikonon.

Kód minta:

import matplotlib.pyplot as plt

import datetime as dt

filename = 'test.csv'

def read\_temp\_raw(file\_name):

    f = open(file\_name, 'r')

    lines = f.readlines()

    f.close()

    return lines

def read\_data(lines, delimiter=','):

    date = []

    tc = []

    for line in lines[1:]:

        a = line.strip().split(',')

        dd = # alakitsd at az 'a' lista elso elemet (ido) datetime formatumba a strptime metodussal felhasznalva a string alakot '%Y-%m-%d %H:%M:%S'

        # add a 'date' listahoz a dd erteket az append metodussal

        # add az 'a' masodik elemet (celsius fok) a tc listahoz az append metodussal. Hasznald a float parancsot, hogy a stringet szamma alakitsd

    return date, tc

lines = # olvasd ki a filename nevu fajl tartalmat a read\_temp\_raw fuggvennyel

d,t = # valaszd szet a datumot es a homersekletet a read\_data fuggvennyel

# plottold a t valozot a d fuggvenyeben, a marker tipusa legyen korocske vonallal: 'o-'

plt.gcf().autofmt\_xdate()

plt.show()

import matplotlib.pyplot as plt

import os, glob, time

import datetime as dt

os.system('modprobe w1-gpio')

os.system('modprobe w1-therm')

base\_dir = '/sys/bus/w1/devices/'

device\_folder = glob.glob(base\_dir + '28\*')[0]

device\_file = device\_folder + '/w1\_slave'

def read\_temp\_raw(file\_name):

    f = open(file\_name, 'r')

    lines = f.readlines()

    f.close()

    return lines

def read\_temp(file\_name):

    lines = read\_temp\_raw(file\_name)

    now = dt.datetime.today().strftime('%Y-%m-%d %H:%M:%S')

    while lines[0].strip()[-3:] != 'YES':

        time.sleep(0.2)

        lines = read\_temp\_raw(file\_name)

    equals\_pos = lines[1].find('t=')

    if equals\_pos != -1:

        temp\_string = lines[1][equals\_pos+2:]

        temp\_c = float(temp\_string) / 1000.0

        temp\_f = temp\_c \* 9.0 / 5.0 + 32.0

        return temp\_c, temp\_f, now

def prepare\_data(date, tc, dplot, tplot, maxlen=20):

    dd = # alakitsd at az 'a' lista elso elemet (ido) datetime formatumba a strptime metodussal felhasznalva a string alakot '%Y-%m-%d %H:%M:%S'

    # add a dplot listahoz a dd erteket az append metodussal

    # add a tplot listahoz a tc erteket az append metodussal

    if len(dplot) > maxlen:

        dplot.pop(0)

        tplot.pop(0)

    return dplot, tplot

max\_count = 20

dplot = [dt.datetime.now()]

tplot = [23.6]

plt.ion()

figure, ax = plt.subplots(figsize=(8,6))

line1, = # plottold a tplotot a dplot fuggvenyeben  'o-' markert hasznalva. Itt az ax.plot fuggvenyt kell majd hasznalni

plt.title(..........,fontsize=25)  # adj cimet a plotnak string formaban a ..... helyen

plt.xlabel("Time",fontsize=18)

plt.ylabel("Temperature (C)",fontsize=18)

plt.grid(True)

while True:

    tc, tf, ts = # olvasd ki a homersekletet a device\_file fajlbol a read\_temp fuggveny segitsegevel

    print(ts)

    dplot, tplot = # keszitsd elo a plottolni kivant listakat (dplot, tplot) a prepare\_data fugvennyel. A maxlen erteke legyen max\_count

    line1.set\_xdata(dplot)

    line1.set\_ydata(tplot)

    ax.set\_ylim(min(tplot)\*0.99,max(tplot)\*1.01) # +1 to avoid singular transformation warning

    ax.set\_xlim(min(dplot),max(dplot))

    figure.canvas.draw()

    figure.canvas.flush\_events()

    plt.gcf().autofmt\_xdate()

    time.sleep(1)