```python
# Simple program for automatized measurements
# ver. 20.3.2020
# Frantisek Kundracik

import serial #pip install pyserial  - in command prompt
import time
import serial.tools.list_ports
import tkinter as tk  #pip install tk  - in command prompt
from tkinter import filedialog
import os
import sys
import keyboard #pip install keyboard  - in command prompt

cwd=os.path.dirname(__file__)+'\\'

def setIpanoPosition(vyska,azimut):
    global seripano
    try:
        command=':01SSL+%s%s#' %
        (str((int)(100*vyska)).zfill(5),str((int)(100*azimut)).zfill(5))
        command=command.encode() #prerobit na postupnot 8-bit znakov
        #print(command)
        seripano.write(command) #move to the given position
        seripano.read(19) #odpoveď
        time.sleep(0.1) #aby sa montáž stihla rozbehnúť
        seripano.write(b':01GAS#') #get current position and state
        answer=seripano.read(19)
        while answer[17]== ord('1'): #kód jednotky - je v pohybe
            seripano.write(b':01GAS#') #get current position and state
            answer=seripano.read(19)
        return(answer.decode)
    except:
        return('Error')

def restoreConnection():
    global seripano
    global serhead
    print('\a')
    print("USB connection is lost. Reconnect and press ENTER...")
    input('')
    print('Restoring connection to iPANO mount...')
    seripano.close()
    seripano = serial.Serial(myipanoportdevice,115200, timeout=1)  # open serial port
    print('OK')
    print('Restoring connection to the measurement head...')
    serhead.close()
    serhead = serial.Serial(myheadportdevice,115200, timeout=1)  # open serial port
    time.sleep(5) #wait for the reset
    print('Initializing the measurement head...')
    headCommand('RFL0XXXX')
    headCommand('RFL1XXXX')
    print('OK')
    readManualCommands()


def headCommand(cmd):
    answer = b'Error'
    try:
        while(serhead.in_waiting>0):
            serhead.read(1)

        serhead.write(cmd.encode())
        while(serhead.in_waiting<8):
            pass
        answer=serhead.read(8)
        time.sleep(0.05)   #wait for rest of data
        while(serhead.in_waiting>0):
            serhead.read(1)
    except:
        pass
    return answer.decode()

```

```
72  def manualCommands():
73      global cwd
74      print('Send commands to the mount. Write "q" for finish:')
75      subor = open(cwd+'running_commands.txt', 'w')
76      command=input('> ')
77      while command != 'q':
78          while len(command)<8:
79              command=command+'X'
80          #print(command)
81          subor.write(command+'\n')
82          print('>',headCommand(command))
83          command=input('> ')
84      subor.close()
85
86  def readManualCommands():
87      global cwd
88      print('Executing manual commands...')
89      subor = open(cwd+'running_commands.txt', 'r')
90      command = subor.readline().strip()
91      while command != '':
92          headCommand(command)
93          command = subor.readline().strip()
94      subor.close()
95
96
97
98  root = tk.Tk()
99  root.withdraw()  #block main GUI window
100
101 myipanoportdevice=''
102 myheadportdevice=''
103 ports = serial.tools.list_ports.comports(include_links=False)
104 print('Ports found:')
105 for port in ports:
106     print(port.device,'\t',port.description)
107 print('Searching for iPANO mount...')
108 seripano=False
109 iPANOfound=False
110 serhead=False
111 headfound=False
112 for port in ports :
113     strport=port.description
114     if strport.find('luetooth')<0:
115         try:
116             print('Checking '+port.device)
117             seripano = serial.Serial(port.device,115200, timeout=1)  # open serial
                port
118             seripano.write(b':01INF#')
119             s=seripano.read(11)
120             if s==b':10INF3600#':
121                 print('iPANO mount found on '+port.device)
122                 myipanoportdevice=port.device
123                 iPANOfound=True
124                 break
125         except:
126             print('Not found')
127         if seripano:
128             seripano.close()
129 if iPANOfound==False:
130     print('iPANO mount not found')
131     #quit()
132 print('Searching for the measurement head...')
133 for port in ports:
134     strport=port.description
135     if strport.find('luetooth')<0:
136         try:
137             print('Checking '+port.device)
138             serhead = serial.Serial(port.device,115200, timeout=1)  # open serial port
139             time.sleep(5) #wait for the reboot of the device
140             serhead.write(b'IDNXXXXX')
141             s=serhead.read(8)
142             #print(s)
```

```
            if s==b'SKY-SCAN':
                print('Measurement head found on '+port.device)
                headfound=True
                myheadportdevice=port.device
                break
        except:
            print('Not found')
        if serhead:
            serhead.close()

if headfound==False:
    print('Measurement head mount not found')
    #quit()

print('Initializing the measurement head...')
headCommand('RFL0XXXX')
headCommand('RFL1XXXX')

filename_out=''
continue_meas=False
if os.path.isfile(cwd+'running.txt'):
    answer=input('Last measurement was stopped unexpectedly. Continue? (y/n): ')
    if(answer!='n'):
        continue_meas=True


if(continue_meas==False):
    print('Moving to the start position...')
    setIpanoPosition(0,0)

continue_no=0
if(continue_meas==True):
    file_run=open(cwd+"running.txt","r")
    file_positions_path=file_run.readline().strip()
    file_measurements_path=file_run.readline().strip()
    filename_out=file_run.readline().strip()
    continue_no=int(file_run.readline().strip())
    file_run.close()
else:
    print('Select the file containing head positions...')
    file_positions_path = filedialog.askopenfilename(parent=root)  #parent=root -
    dialog on top
    print('Select the file containing measurement commands...')
    file_measurements_path = filedialog.askopenfilename(parent=root)

print('Reading list of head positions from "',file_positions_path,'" file...')
positions=[]
try:
    subor = open(file_positions_path, 'r')
    riadok = subor.readline()
    while riadok != '':
        cislo = riadok.split("\t")
        vyska=float(cislo[0].strip())
        azimut=float(cislo[1].strip())
        positions.append((vyska,azimut))
        riadok = subor.readline()
    subor.close()
except FileNotFoundError:
    print("Configuration file '",file_positions_path,"' not found...")
    quit()
print(len(positions),' positions read')

print('Reading list of measurement commands "',file_measurements_path,'" file...')
headline="";
commands=[]
try:
    subor = open(file_measurements_path, 'r')
    headline=subor.readline()
    headline=headline.strip()
    riadok = subor.readline()
    riadok = riadok.strip()
    while riadok != '':
```

```
            commands.append(riadok)
            #print('---',riadok,'---')
            riadok = subor.readline()
            riadok = riadok.strip()
        subor.close()
    except FileNotFoundError:
        print('Configuration file "',file_measurements_path,'" not found...')
        quit()
    print(len(commands),' commands read')

    if(continue_meas==False):
        print('Choose the name of the output-file...')
        filename_out = filedialog.asksaveasfilename(parent=root)
        if filename_out is None: # asksaveasfile return `None` if dialog closed with
        "cancel".
            filename_out=cwd+'measurement.txt'
        fileout=open(filename_out,"w")
        fileout.close() #clear the content of the file

    if(continue_meas==True):
        readManualCommands()
    else:
        manualCommands()



    print('---------- Start of measurements ---------')
    print('Press Ctrl to pause the measurement')
    print('finished', 'vyska', 'azimut', headline)

    if(continue_meas==False):
        buf='z.angle\tazimuth\t%s\n' % (headline)
        fileout=open(filename_out,"a")
        fileout.write(buf)
        fileout.close()

    index=0;
    answer=''
    for (vyska, azimut) in positions:
        if(index<continue_no):
            index+=1
            continue
        answer=setIpanoPosition(vyska,azimut)
        if(answer=='Error'):
            restoreConnection()
            answer=setIpanoPosition(vyska,azimut)

        print(int(100.0*(index+1)/len(positions)),'%',vyska, azimut,end='\t')
        buf='%.2f\t%.2f\t' % (90.0-vyska,azimut)

        for command in commands:
            #check for key pressed: loop until another key pressed
            if keyboard.is_pressed('Ctrl'):
                print('\a')
                print('Paused, press Shift to continue...')
                keyboard.wait('Shift')
                print('Running again')
            answer=headCommand(command)
            if answer=='Error':
                break
            if answer.find('SVT')==0:
                value=int(answer[3:8])
                #print(answer[3:8])
                value=value/10000
                print('%.4f' % value,'\t',end='')
                buf=buf+'%.4f\t' % (value)

        if(answer=='Error'):
            restoreConnection()
            buf='%.2f\t%.2f\t' % (90.0-vyska,azimut)
            print(int(100.0*(index+1)/len(positions)),'%',vyska, azimut,end='\t')
            for command in commands:
```

```
285                 answer=headCommand(command)
286                 if(answer=='Error'):
287                     print('\a\nRepeating problem with USB connection was detected. Check
                        the cables and run the program again.');
288                     sys.exit()
289                 if answer.find('SVT')==0:
290                     value=int(answer[3:8])
291                     #print(answer[3:8])
292                     value=value/10000
293                     print('%.4f' % value,'\t',end='')
294                     buf=buf+'%.4f\t' % (value)
295 
296         if(answer!='Error'):
297             buf=buf+'\n'
298             print('')
299             fileout=open(filename_out,"a")
300             fileout.write(buf)
301             fileout.close()
302             index+=1
303             #Značka kam sme sa dostali
304             subor=open(cwd+'running.txt','w')
305             subor.write(file_positions_path+'\n')
306             subor.write(file_measurements_path+'\n')
307             subor.write(filename_out+'\n')
308             subor.write(str(index)+'\n')
309             subor.close()
310 
311 
312     print('')
313     print('---------- End of measurements ---------')
314 
315     if os.path.isfile(cwd+'running.txt'):
316         os.remove(cwd+'running.txt')
317     if os.path.isfile(cwd+'running_commands.txt'):
318         os.remove(cwd+'running_commands.txt')
319 
320     print('Closing filters...')
321     print(headCommand('SFL000XX'))
322     print(headCommand('SFL100XX'))
323 
324     print('Moving to the initial position...')
325     setIpanoPosition(0,0)
326 
327     print('Initializing the measurement head...')
328     print(headCommand('RFL0XXXX'))
329     print(headCommand('RFL1XXXX'))
330 
331 
332     seripano.close()              # close port
333     serhead.close()
334 
```