1 Logging Data From Serial Ports Exercise

Exercise 1.
Import the `serial` module and open the serial port with the appropriate parameters.

Exercise 2.
Get a reading from the temperature probe.

Exercise 3.
Add a date and time reading to your output, using sensible choices for format, timezone, etc.

Exercise 4.
Add a loop to your code to continuously log the reading and time. What would be a good exit condition? Hint: try `dir(serial.Serial)` to see what methods might be of use.

Exercise 5.
Rewrite your code to use `readline()`.

Exercise 6.
Alter your code to write the data out to a file.

Solution 1.

```
#!/usr/bin/env python
import serial

ser = serial.Serial(
    port='/dev/ttyUSB0',
    baudrate=9600,
    bytesize=serial.EIGHTBITS,
    parity=serial.PARITY_NONE,
    stopbits=serial.STOPBITS_ONE
)
```

Solution 2.

`print(ser.read(size=8))`

”8” here is specific to the Papouch thermometer device.

Solution 4.

Several ways, but the simplest is:

```
while ser.isOpen():
    datastring = ser.read(size=8)
    print datetime.utcnow().isoformat(), datastring
```

Solution 5.

```
import io
...
sio = io.TextIOWrapper(io.BufferedRWPair(ser, ser, 1), encoding='ascii', newline='\r')

while ser.isOpen():
    datastring = sio.readline()
    print datetime.utcnow().isoformat(), datastring
```

Solution 6.

```
#!/usr/bin/env python
'''This version of the readserial program demonstrates
using python to write an output file'''

from datetime import datetime
import serial
import io

outfile='/tmp/serial-temperature.tsv'

ser = serial.Serial(
   port='/dev/ttyUSB0',
   baudrate=9600,
)

sio = io.TextIOWrapper(
   io.BufferedRWPair(ser, ser, 1),
   encoding='ascii', newline='\r'
)
sio._CHUNK_SIZE =1

with open(outfile,'a') as f: #appends to existing file
   while ser.isOpen():
      datastring = sio.readline()
      #\t is tab; \n is line separator
      f.write(datetime.utcnow().isoformat() + '\t' + datastring + '\n')
      f.flush() #included to force the system to write to disk

ser.close()
```

(see `python/exercises/example_code/ldfsp.py` in your `ncas-isc` checkout)

2 Writing and Plotting NetCDF files Exercise

Exercise 7.
Write a function to convert the time as written in your datafile and return a Python `datetime` object.

p

Exercise 8.
Write a function to convert the temperature as written in your datafile and return a `float` in Kelvin.

$T_K = T_C + 273.15$

Exercise 9.
Read your datafile into Python using the `csv` module such that you end up with list object(s) containing floating-point temperature in K and timestamps as Python `datetime` objects.
see: https://docs.python.org/3/library/csv.html

Exercise 10.

- Create a `Dataset` (use the format `NETCDF4_CLASSIC`)
- Convert your time series to a suitable CF-compliant series
- Create a suitable `Dimension` for your time series
- Create `Variable` objects for Temp and Time using appropriate units etc.
- Assign appropriate metadata to the Temp `Variable` and and the `Dataset`
- Add your time series and temp values to the `Dataset`
- Close and write your `Dataset`. Test that it parses correctly with `ncdump`

Bonus Exercise 11.
You can do a quick-and-dirty plot with `ncview`:

```
ncview sensor_data.nc
```

This isn't publication quality. Produce a line plot of temperature vs time using `matplotlib` and reading the data from your NetCDF file.

Bonus Exercise 12.
"CIS is an open source command-line tool for easy collocation, visualization, analysis, and comparison of diverse gridded and ungridded datasets used in the atmospheric sciences" It is based on python. Homepage: http://www.cistools.net/

```
cis plot temp:sensor_data.nc --xaxis time --yaxis temp \
    --title "Papouch Thermometer Data, 2017-02-22, UoL PRD" --xstep "0.010416"  \
    --output sensor_data_sample.svg
```

Experiment with CIS.

Solution 7.

```
def convert_time(tm):
    tm = datetime.strptime(tm, "%Y-%m-%dT%H:%M:%S.%f")
    return tm
```

`strptime` is the opposite of `strftime` that we used earlier.

Solution 8.

```
def convert_temp(temp):
    value = temp.strip("+").strip("C").lstrip("0")
    return float(value) + 273.15
```

Solution 9.

```
infile='sample-serial-temperature-2h.tsv' #Or whatever your infile is called
outfile='sensor-data.nc'
from csv import reader

# Parse the data into python lists
times = []
temps = []

#open infile and read data into lists
with open(infile, 'rb') as tsvfile:
   tsvreader = reader(tsvfile, delimiter='\t')
   for row in tsvreader:
      times.append(convert_time(row[0]))
      temps.append(convert_temp(row[1]))
```

Solution 10.

See: `python/exercises/example_code/write_sensor_data_to_netcdf.py` in your `ncas-isc` checkout.

Solution 11.

See: `python/exercises/example_code/plot-netcdf.py`

Solution 12.

See: `python/presentations/logging-data-from-serial-ports/sensor_data_sample.svg`