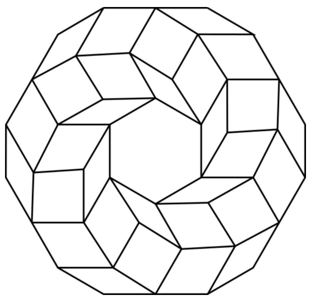

Reading and writing text files

Developed by Onno Ebbens

Reading and writing text files can be very useful. There are many ways to read and write files from all kinds of formats. In this notebook we will discuss reading and writing text files.

Table of content:

1. reading text files

There are several ways to read a text file. The most basic way is to use the function open(). This function opens a text file and returns a handle to that text file. If you use the open() function you need to specify whether you want to read ('r') or write ('w') to that text file.

We use a 'with' statement to read the file. This means that everything after the with statement with same indentation will be executed and then the document is closed again. If we would not do this, the document will be opened continuously by python which can be the cause of some unsuspected errors.

with the readlines() method we read the text file as a list of strings.

In [ ]:

```
#this lines creates a handle to the textfile
with open(r'data\InhabitantsNL.txt', 'r') as txtfile:
    lines = txtfile.readlines()
lines
```

Exercise 1: Open the 'InhabitantsNL.txt' with a textfile editor and compare your results with the list shown above. What does the '\t' means in your textfile? What does the '\n' means?

There are several methods to read textfiles. One that is often used is the readline() method. This method reads one line at a time which can be usefull if you only want to read the top part of a big text file.

In [ ]:

```
with open(r'data\InhabitantsNL.txt', 'r') as txtfile:
    first_line  = txtfile.readline()
    second_line = txtfile.readline()
print(first_line)
print(second_line)
```

Now if you want to do something with this text file you need to convert the text strings to something you can use. For example if you want to plot this data you can do the following.

In [ ]:

```
#We want two lists with numbers that we can plot.
#We can make these with a for loop.
year = []
inhab = []
#We loop over the lines in the textfilelist (we skip the first line because it does not
 contain data)
for line in lines[1:]:
    #we know that the first 4 characters are the year, so we slice this and convert to
 an integer
    year.append(int(line[:4]))
    #we also know where we can find the number of inhabitants, so we slice and convert
 to a float
    inhab.append(float(line[6:12]))
```

In [ ]:

```
#now we have to lists
print(year)
print(inhab)
```

In [ ]:

```
#now we can make a plot
import matplotlib.pyplot as plt
%matplotlib inline
ax = plt.plot(year,inhab)
```

2. Advanced textfile reading

The example above shows the most basic way of reading files. With this functions you can read any file that contains text. For textfiles with a certain structure there are easier ways to read them. Below we show two from the numpy and the pandas module.

In [ ]:

```
#From the numpy module you can direcly read text as a numpy array
import numpy as np
#you only need to specify how many rows you skip to see the data
#in this case you skip the first row because it only contains the column names
TextFileArray = np.loadtxt('data\InhabitantsNL.txt',skiprows=1)
```

Exercise 2: Plot the data that is read with the np.loadtxt() function

Still, we haven't fully optimized reading a textfile. We lost the information about the columns names when we read the data with np.loadtxt(). Wouldn't it be nice if we could use all the information in the textfile?

Yes! That would be amazing:) With Pandas you can do exactly that!

In [ ]:

```
import pandas as pd
#With pandas you can read files as a dataframe
#The only thing you need to specify is the delimiter. In this case the numbers are deli
mited with a tab ('\t')
TextFileDF = pd.read_csv('data\InhabitantsNL.txt', delimiter='\t')
```

In [ ]:

```
#This is what a dataframe looks like
TextFileDF
```

Exercise 3: What happens if you use plt.plot() on a Dataframe? What has happened? How can you make a proper plot?

In [ ]:

```
plt.plot(TextFileDF)
```

3. writing text files

Just like reading textfiles there is also a basic way to write textfiles.

In [ ]:

```
with open(r"output\output.txt", "w") as txtfile:
    txtfile.write('hello world')
```

Exercise 4: How do you write the TextFileList (the one you read before) to the "output.txt" file?

4. Advanced textfile writing

Off course you can use advanced writing functions if you want to save different data types

Exercise 5: How would you write a numpy array to a textfile?

Exercise 6: How do you write a pandas dataframe to a textfile?

5. final exercises

Exercise 7: Write a function that can read a file similar to "InhabitantsNL.txt" and make a plot(). You can assume that the left column contains years and the right column contains data.