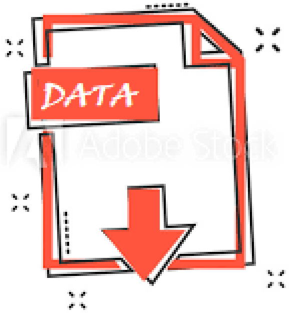
File Input/Output

File IO in Python

Often you want to put calculated or generated data in a file so that the results are still available later.

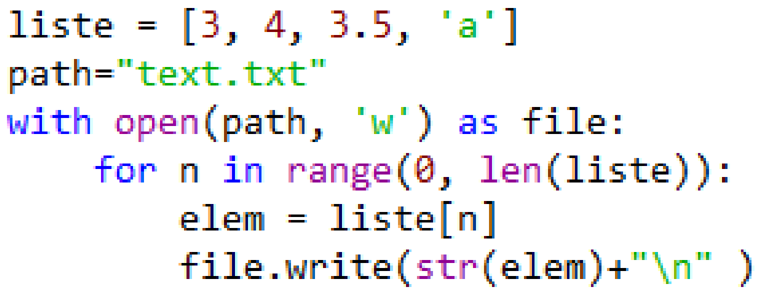
Or, you may already have a file with data that you want to analyze.



In Python, storing data in files and reading date from files is very easy.

Write a list of numbers to a file

Write a list of numbers to a file



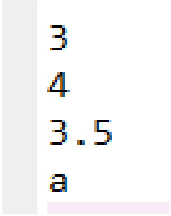
```
liste = [3, 4, 3.5, 'a']
path="text.txt"
with open(path, 'w') as file:
    for n in range(0, len(liste)):
        elem = liste[n]
        file.write(str(elem)+"\n" )
```

Open file under the path in *write* mode

Convert number to text (str) -> only texts can be written to files , end each line with “\n” -> new line

The file is automatically closed at the end

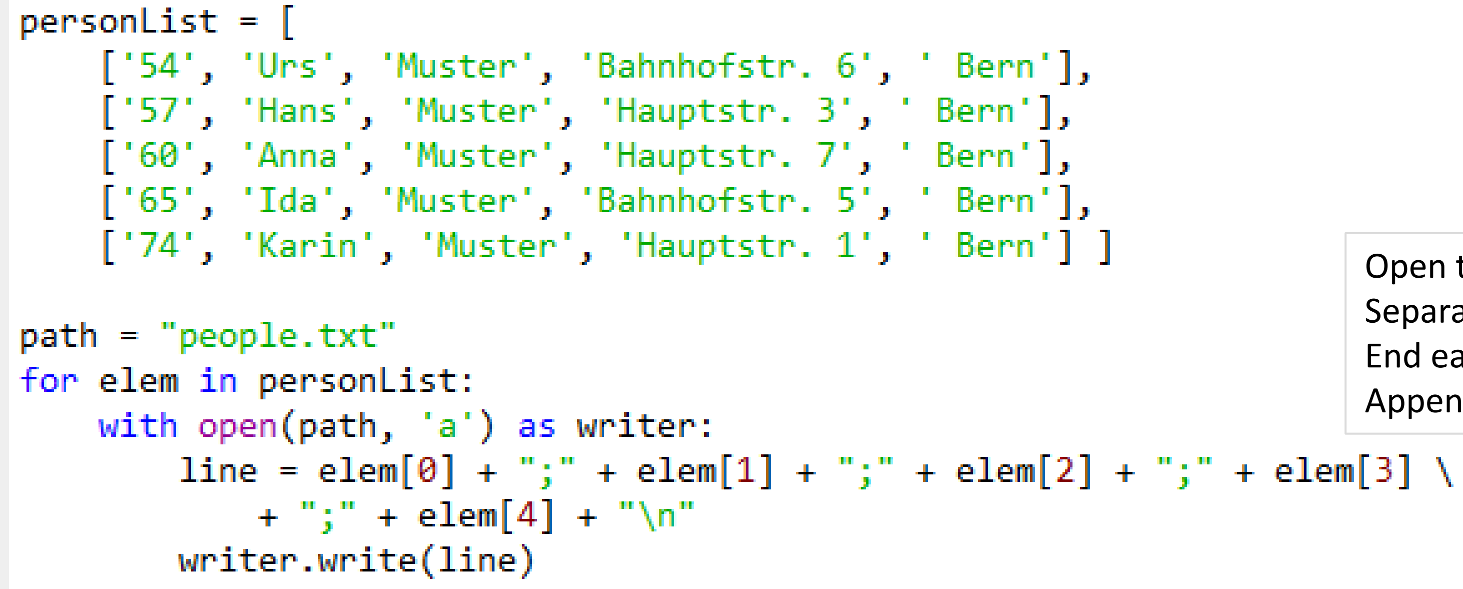
File content



```
3
4
3.5
a
```

Write some text into a file

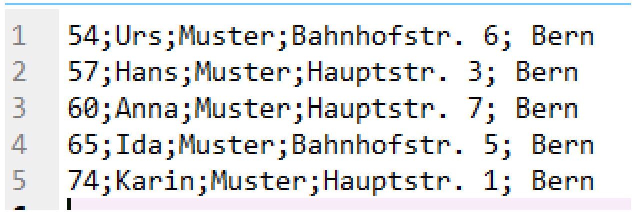
Write personal data to a file



```
personList = [
    ['54', 'Urs', 'Muster', 'Bahnhofstr. 6', ' Bern'],
    ['57', 'Hans', 'Muster', 'Hauptstr. 3', ' Bern'],
    ['60', 'Anna', 'Muster', 'Hauptstr. 7', ' Bern'],
    ['65', 'Ida', 'Muster', 'Bahnhofstr. 5', ' Bern'],
    ['74', 'Karin', 'Muster', 'Hauptstr. 1', ' Bern'] ]

path = "people.txt"
for elem in personList:
    with open(path, 'a') as writer:
        line = elem[0] + ";" + elem[1] + ";" + elem[2] + ";" + elem[3] \
            + ";" + elem[4] + "\n"
        writer.write(line)
```

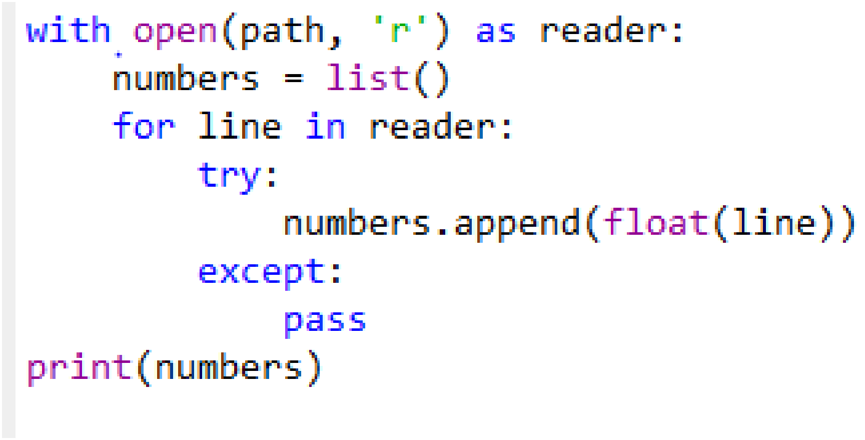
Open the file under the path in *append* mode
Separate individual items by ";"
End each line with a line break -> "\n“
Append line to the file

The content of the person.txt file:



```
54;Urs;Muster;Bahnhofstr. 6; Bern
57;Hans;Muster;Hauptstr. 3; Bern
60;Anna;Muster;Hauptstr. 7; Bern
65;Ida;Muster;Bahnhofstr. 5; Bern
74;Karin;Muster;Hauptstr. 1; Bern
```

Read numbers from a file



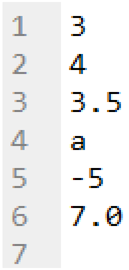
```python
with open(path, 'r') as reader:
    numbers = list()
    for line in reader:
        try:
            numbers.append(float(line))
        except:
            pass
print(numbers)
```

Open File for reading
Read line by line
Convert and output read texts (if possible) in whole numbers.

Ignore exceptions

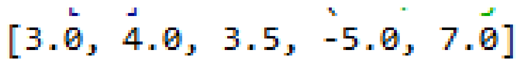
Print read numbers

File content



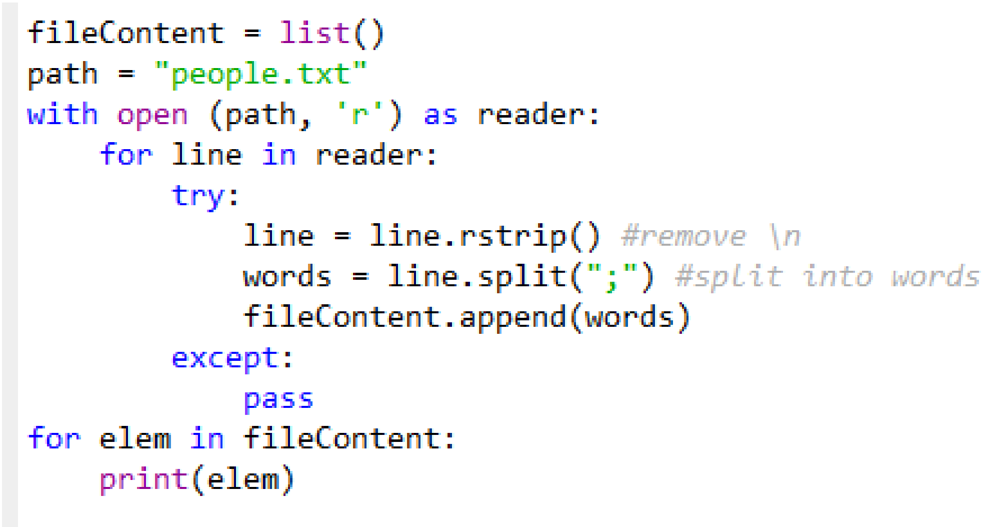
```
3
4
3.5
a
-5
7.0

```

Output



```
[3.0, 4.0, 3.5, -5.0, 7.0]
```

Read text from a file



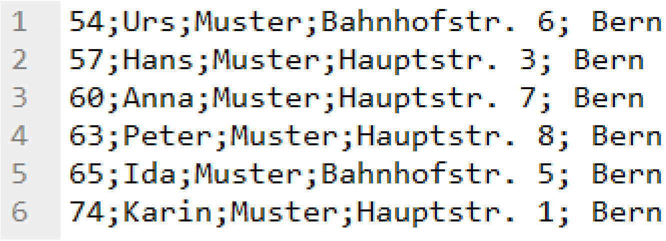
```python
fileContent = list()
path = "people.txt"
with open (path, 'r') as reader:
    for line in reader:
        try:
            line = line.rstrip() #remove \n
            words = line.split(";") #split into words
            fileContent.append(words)
        except:
            pass
for elem in fileContent:
    print(elem)
```

Open file at path for reading

Remove newline

Words are separated by ";"
Append words to list.

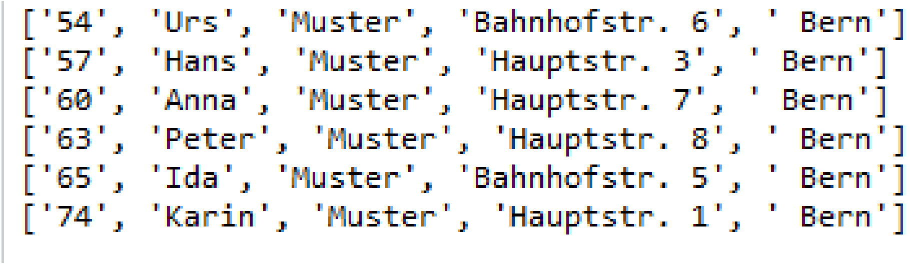
File content



```
54;Urs;Muster;Bahnhofstr. 6; Bern
57;Hans;Muster;Hauptstr. 3; Bern
60;Anna;Muster;Hauptstr. 7; Bern
63;Peter;Muster;Hauptstr. 8; Bern
65;Ida;Muster;Bahnhofstr. 5; Bern
74;Karin;Muster;Hauptstr. 1; Bern
```

people.txt

Output



```
['54', 'Urs', 'Muster', 'Bahnhofstr. 6', ' Bern']
['57', 'Hans', 'Muster', 'Hauptstr. 3', ' Bern']
['60', 'Anna', 'Muster', 'Hauptstr. 7', ' Bern']
['63', 'Peter', 'Muster', 'Hauptstr. 8', ' Bern']
['65', 'Ida', 'Muster', 'Bahnhofstr. 5', ' Bern']
['74', 'Karin', 'Muster', 'Hauptstr. 1', ' Bern']
```

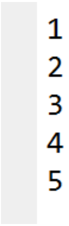
print fileContent

Append to a file with ‘a’

```
liste = ['a','b','c','d']

path = "test.txt"
with open(path,'a') as file:
    for ele in liste:
        file.write(ele+"\n")
```

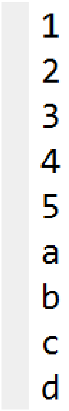
Open file for appending

Initial file content



```
1
2
3
4
5
```

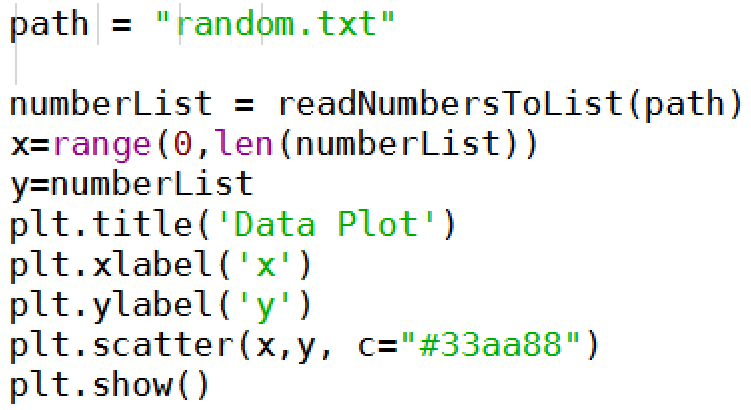
File content after appending



```
1
2
3
4
5
a
b
c
d
```

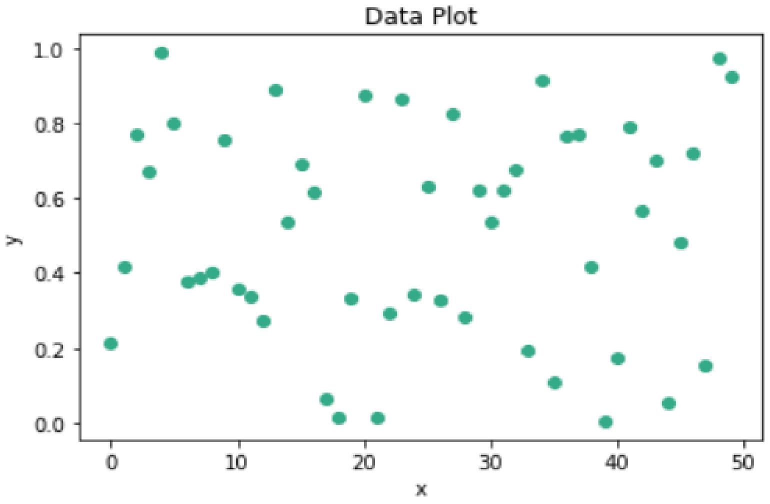
Plot read numbers

Name of your file



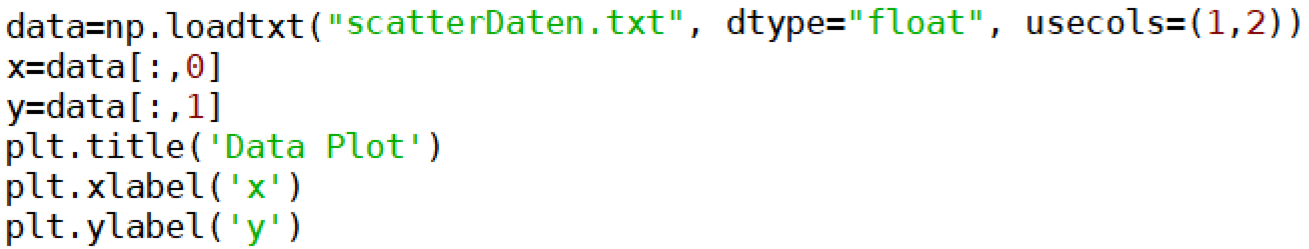
```
path = "random.txt"

numberList = readNumbersToList(path)
x=range(0,len(numberList))
y=numberList
plt.title('Data Plot')
plt.xlabel('x')
plt.ylabel('y')
plt.scatter(x,y, c="#33aa88")
plt.show()
```

Output

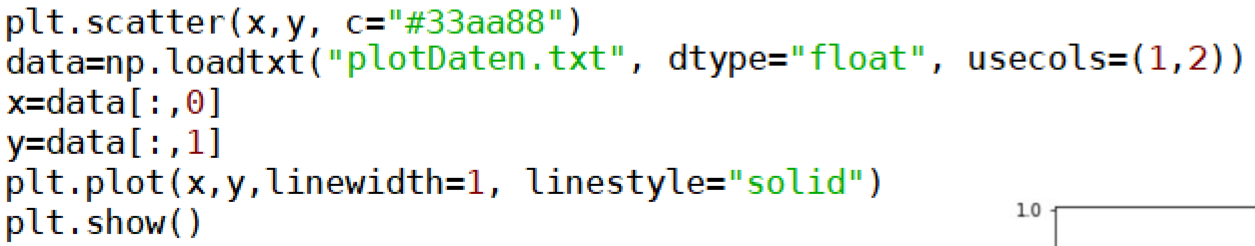


Read DataFiles: loadtxt

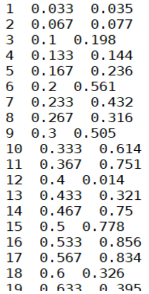
In the numpy[1] modul of Python there is another easy way to read data files



```
data=np.loadtxt("scatterDaten.txt", dtype="float", usecols=(1,2))
x=data[:,0]
y=data[:,1]
plt.title('Data Plot')
plt.xlabel('x')
plt.ylabel('y')
```

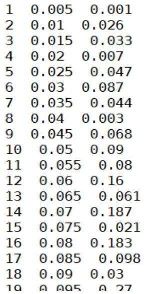


```
plt.scatter(x,y, c="#33aa88")
data=np.loadtxt("plotDaten.txt", dtype="float", usecols=(1,2))
x=data[:,0]
y=data[:,1]
plt.plot(x,y,linewidth=1, linestyle="solid")
plt.show()
```



```
1  0.033  0.035
2  0.067  0.077
3  0.1  0.198
4  0.133  0.144
5  0.167  0.236
6  0.2  0.561
7  0.233  0.432
8  0.267  0.316
9  0.3  0.505
10  0.333  0.614
11  0.367  0.751
12  0.4  0.014
13  0.433  0.321
14  0.467  0.75
15  0.5  0.778
16  0.533  0.856
17  0.567  0.834
18  0.6  0.326
19  0.633  0.395
```

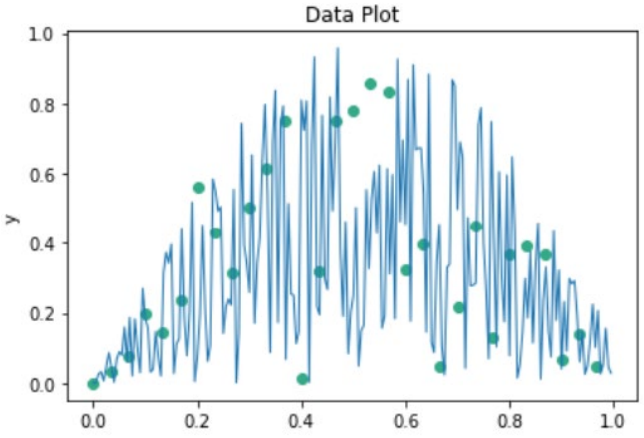
scatterDaten.txt



```
1  0.005  0.001
2  0.01  0.026
3  0.015  0.033
4  0.02  0.007
5  0.025  0.047
6  0.03  0.087
7  0.035  0.044
8  0.04  0.003
9  0.045  0.068
10  0.05  0.09
11  0.055  0.08
12  0.06  0.16
13  0.065  0.061
14  0.07  0.187
15  0.075  0.021
16  0.08  0.183
17  0.085  0.098
18  0.09  0.03
19  0.095  0.27
```

plotDaten.txt

OutputPlot



[1] Numpy is the Python module for scientific computing

File manipulation operations

- https://www.w3schools.com/python/python_file_handling.asp

- https://phoenixnap.com/kb/file-handling-in-python#deleting-files-in-python

- https://www.analyticsvidhya.com/blog/2021/10/introduction-to-file-operations-in-python/