UNIVERSITÄT
ZU KÖLN

SPRACHVERARBEITUNG: ÜBUNG

SoSe 2024

Janis Pagel

Institut für Digital Humanities

2024-04-16

01

PYTHON CRASH COURSE: PART II

Reading and Writing Files I

data.txt

```
William Shakespeare
As You Like It

ACT 1
Scene 1

Enter Orlando and Adam.
ORLANDO
As I remember, Adam, it was upon this
fashion bequeathed me by will but poor a thousand
crowns, and, as thou sayst, charged my brother on
```

- `with ... as ...:` opens the file in a separated environment, so you don't need to take care of closing the file
- `file_object.read()` returns the file content as a string

```
with open("data.txt", "r") as file_object:
      file_read = file_object.read()
print(file_read)
print(file_read.split("\n"))

> William Shakespeare
> As You Like It
>
> ACT 1
> Scene 1
>
> Enter Orlando and Adam.
> ORLANDO
> As I remember, Adam, it was upon this
> fashion bequeathed me by will but poor a
                      thousand
> crowns, and, as thou sayst, charged my
                      brother on

> ['William Shakespeare', 'As You Like It',
                      '', 'ACT 1', 'Scene 1'
                      , '', 'Enter Orlando
                      and Adam.', 'ORLANDO',
                       'As I remember, Adam,
                       it was upon this', '
                      fashion bequeathed me
                      by will but poor a
                      thousand', 'crowns,
                      and, as thou sayst,
                      charged my brother on'
```

Reading and Writing Files II

data.txt

```
William Shakespeare
As You Like It

ACT 1
Scene 1

Enter Orlando and Adam.
ORLANDO
As I remember, Adam, it was upon this
fashion bequeathed me by will but poor a thousand
crowns, and, as thou sayst, charged my brother on
```

- readlines() directly splits the file content by newline and returns a list, but preserves the newlines

```
with open("data.txt", "r") as file_object:
      file_read = file_object.readlines()
print(file_read)

> ['William Shakespeare\n', 'As You Like It\
                       n', '\n', 'ACT 1\n', '
                       Scene 1\n', '\n', '
                       Enter Orlando and Adam
                       .\n', 'ORLANDO\n', 'As
                        I remember, Adam, it
                       was upon this\n', '
                       fashion bequeathed me
                       by will but poor a
                       thousand\n', 'crowns,
                       and, as thou sayst,
                       charged my brother on'
                       ]
```

Reading and Writing Files III

data.txt

```
William Shakespeare
As You Like It

ACT 1
Scene 1

Enter Orlando and Adam.
ORLANDO
As I remember, Adam, it was upon this
fashion bequeathed me by will but poor a thousand
crowns, and, as thou sayst, charged my brother on
```

- `open(..., "w")` writes to a file, creates it if it doesn't exists yet and overwrites it if it does exist (without asking for confirmation!!!)
- `"sep".join()` takes a list as argument and returns a string with `"sep"` as the separator

```
with open("data.txt", "r") as file_object:
      file_read = file_object.read()
file_split = file_read.split("\n")
file_sorted = sorted(file_split)
with open("sorted.txt", "w") as file_object:
      file_object.write("\n".join(
                            file_sorted))
```

sorted.txt

```


ACT 1
As I remember, Adam, it was upon this
As You Like It
Enter Orlando and Adam.
ORLANDO
Scene 1
William Shakespeare
crowns, and, as thou sayst, charged my brother on
fashion bequeathed me by will but poor a thousand
```

Classes and Methods

```
class Text:
    # methods in-between __ are built-in and have special functionalities
    # __init__: Is executed when class object is created
    # self refers to the class object (this in Java)
    def __init__(self, path):
        self.path = path
        self.text = self._read_file()

    # Starting a method with _ is a convention that the method is only used class-internally (like "private"
                                                            in Java, but not enforced)
    def _read_file(self):
        with open(self.path, "r") as file_object:
            text = file_object.read()
        return text

    # Functions without a leading underscore are "public" (but again, this is only a convention)
    def count_words(self):
        return len(self.text.split(" "))

    def count_characters(self):
        return len(self.text)

t = Text("data.txt")
print(t.path)
print(t.count_words())
print(t.count_characters())

> data.txt
> 33
> 220
```

pandas I

- *pandas* is a Python library for organizing and processing tabular data
- *pandas* stores its data in so-called *dataframes*
- Use *pandas* by importing it

```
import pandas
print(pandas.__version__)

> 2.1.1
```

- Often, *pandas* is imported under the name *pd*

```
import pandas as pd
print(pd.__version__)

> 2.1.1
```

pandas II

- Below is an example dataset with comma-separated values, giving the sepal length/width, the petal length/width and the species of six different iris flowers

data.csv

```
sepal_length,sepal_width,petal_length,petal_width,species
5.1,3.5,1.4,0.2,setosa
4.9,3.0,1.4,0.2,setosa
7.0,3.2,4.7,1.4,versicolor
6.4,3.2,4.5,1.5,versicolor
6.3,3.3,6.0,2.5,virginica
5.8,2.7,5.1,1.9,virginica
```

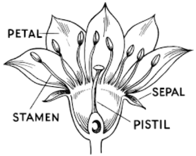


Figure: Flower description

Figure: Iris Versicolor

Figure: Iris Setosa

Figure: Iris Verginica

pandas III

data.csv

```
sepal_length,sepal_width,petal_length,petal_width,species
5.1,3.5,1.4,0.2,setosa
4.9,3.0,1.4,0.2,setosa
7.0,3.2,4.7,1.4,versicolor
6.4,3.2,4.5,1.5,versicolor
6.3,3.3,6.0,2.5,virginica
5.8,2.7,5.1,1.9,virginica
```

```
import pandas as pd
dataframe = pd.read_csv("data.csv", header=0)
print(dataframe)

>    sepal_length  sepal_width  petal_length  petal_width     species
> 0           5.1          3.5           1.4          0.2      setosa
> 1           4.9          3.0           1.4          0.2      setosa
> 2           7.0          3.2           4.7          1.4  versicolor
> 3           6.4          3.2           4.5          1.5  versicolor
> 4           6.3          3.3           6.0          2.5   virginica
> 5           5.8          2.7           5.1          1.9   virginica
```

pandas IV

```
import pandas as pd
dataframe = pd.read_csv("data.csv", header=0)
print(dataframe[["sepal_length", "species"]])
print(dataframe.loc[[0]])
print(dataframe.loc[[0,4], ["petal_width"]])

>    sepal_length     species
> 0           5.1      setosa
> 1           4.9      setosa
> 2           7.0  versicolor
> 3           6.4  versicolor
> 4           6.3   virginica
> 5           5.8   virginica

>    sepal_length  sepal_width  petal_length  petal_width species
> 0           5.1          3.5           1.4          0.2  setosa

>    petal_width
> 0          0.2
> 4          2.5
```

data.csv

```
sepal_length,sepal_width,petal_length,petal_width,species
5.1,3.5,1.4,0.2,setosa
4.9,3.0,1.4,0.2,setosa
7.0,3.2,4.7,1.4,versicolor
6.4,3.2,4.5,1.5,versicolor
6.3,3.3,6.0,2.5,virginica
5.8,2.7,5.1,1.9,virginica
```

pandas V

```
import pandas as pd
dataframe = pd.read_csv("data.csv", header=0)
print(dataframe.species.value_counts())
print(dataframe.species.value_counts(normalize=True))
print(dataframe.groupby("species").mean())

> species
> setosa        2
> versicolor    2
> virginica     2
> Name: count, dtype: int64

> species
> setosa        0.333333
> versicolor    0.333333
> virginica     0.333333
> Name: proportion, dtype: float64

>             sepal_length  sepal_width  petal_length  petal_width
> species
> setosa              5.00         3.25          1.40         0.20
> versicolor          6.70         3.20          4.60         1.45
> virginica           6.05         3.00          5.55         2.20
```

data.csv

```
sepal_length,sepal_width,petal_length,petal_width,species
5.1,3.5,1.4,0.2,setosa
4.9,3.0,1.4,0.2,setosa
7.0,3.2,4.7,1.4,versicolor
6.4,3.2,4.5,1.5,versicolor
6.3,3.3,6.0,2.5,virginica
5.8,2.7,5.1,1.9,virginica
```

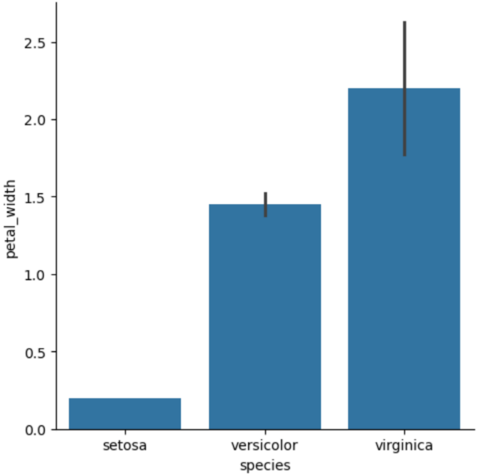
seaborn I

- *seaborn* is a library for plotting data, specifically developed for data science
- seaborn is built ontop of another library, *matplotlib*, which is very powerful and sometimes you will need to look up functions from matplotlib in order to get a certain result in seaborn

data.csv

```
sepal_length,sepal_width,petal_length,petal_width,species
5.1,3.5,1.4,0.2,setosa
4.9,3.0,1.4,0.2,setosa
7.0,3.2,4.7,1.4,versicolor
6.4,3.2,4.5,1.5,versicolor
6.3,3.3,6.0,2.5,virginica
5.8,2.7,5.1,1.9,virginica
```

```python
import seaborn as sns
import pandas as pd
dataframe = pd.read_csv("data.csv", header=0)
sns.catplot(
    data=dataframe, kind="bar",
    x="species", y="petal_width",
    errorbar="sd"
)
```

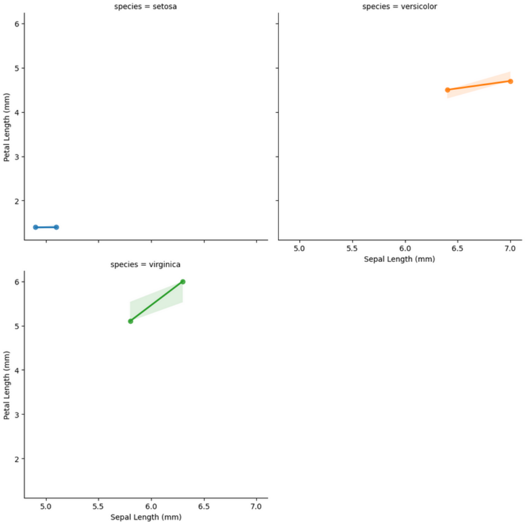


seaborn II

data.csv

```
sepal_length,sepal_width,petal_length,petal_width,species
5.1,3.5,1.4,0.2,setosa
4.9,3.0,1.4,0.2,setosa
7.0,3.2,4.7,1.4,versicolor
6.4,3.2,4.5,1.5,versicolor
6.3,3.3,6.0,2.5,virginica
5.8,2.7,5.1,1.9,virginica
```

```python
import seaborn as sns
import pandas as pd
dataframe = pd.read_csv("data.csv", header=0)
g = sns.lmplot(data=dataframe,
               x="sepal_length",
               y="petal_length",
               col="species",
               hue="species",
               col_wrap=2
               )
g.set_axis_labels("Sepal Length (mm)", "Petal
                            Length (mm)")
```



UNIVERSITY
OF COLOGNE

Janis Pagel
Institut für Digital Humanities

eMail janis.pagel@uni-koeln.de
Website https://janispagel.de

Foto: Gregor Hübl