

Working with Data in Python Cheat Sheet

Reading and writing files

Package/Method Description

File opening modes	Different modes to open files for specific operations.	Syntax: r (reading) w (writing) a (appending) + (updating: read/write) b (binary, otherwise text) 1. 1 1. Examples: with open("data.txt", "r") as file: content = file.read() print(content) with open("output.txt", "w") as file: Copied!
		Syntax: 1. 1 2. 2 3. 3 1. file.readlines() # reads all lines as a list 2. readline() # reads the next line as a string 3. file.read() # reads the entire file content as a string
File reading methods	Different methods to read file content in various ways.	Copied! Example: 1. 1 2. 2 3. 3 4. 4 1. with open("data.txt", "r") as file: 2. lines = file.readlines() 3. next_line = file.readline() 4. content = file.read() Copied!
		Syntax: 1. 1 2. 2 1. file.write(content) # writes a string to the file 2. file.writelines(lines) # writes a list of strings to the file
File writing methods	Different write methods to write content to a file.	Copied! Example: 1. 1 2. 2 3. 3 1. lines = ["Hello\n", "World\n"] 2. with open("output.txt", "w") as file: 3. file.writelines(lines) Copied!
		Syntax: 1. 1 1. for line in file: # Code to process each line Copied! Example: 1. 1 2. 2 1. with open("data.txt", "r") as file: 2. for line in file: print(line) Copied!
Iterating over lines	Opens a file, performs operations, and explicitly closes the file using the close() method.	Syntax: 1. 1 2. 2 1. file = open(filename, mode) # Code that uses the file 2. file.close() Copied!
		Example: 1. 1 2. 2 3. 3 1. file = open("data.txt", "r")

```
2. content = file.read()
3. file.close()
```

Copied!

Syntax:

```
1. 1
```

```
1. with open(filename, mode) as file: # Code that uses the file
```

Opens a file
using a with
block,

Copied!

with open()

ensuring automatic
file closure
after usage.

```
1. 1
```

```
2. 2
```

```
1. with open("data.txt", "r") as file:
2. content = file.read()
```

Copied!

Pandas

Package/Method

Description

Syntax and Code Example

.read_csv() Reads data from a `.CSV` file and creates a DataFrame.

Syntax: `dataframe_name = pd.read_csv("filename.csv")` Example: `df = pd.read_csv("data.csv")`
Syntax:

```
1. 1
```

```
1. dataframe_name = pd.read_excel("filename.xlsx")
```

Copied!

.read_excel() Reads data from an Excel file and creates a DataFrame.

Example:

```
1. 1
```

```
1. df = pd.read_excel("data.xlsx")
```

Copied!

Syntax:

```
1. 1
```

```
1. dataframe_name.to_csv("output.csv", index=False)
```

Copied!

.to_csv()

Writes DataFrame to a CSV file.

Example:

```
1. 1
```

```
1. df.to_csv("output.csv", index=False)
```

Copied!

Syntax:

```
1. 1
```

```
2. 2
```

```
1. dataframe_name["column_name"] # Accesses single column
2. dataframe_name[["column1", "column2"]]] # Accesses multiple columns
```

Copied!

Access Columns Accesses a specific column using `[]` in the DataFrame.

Example:

```
1. 1
```

```
2. 2
```

```
1. df["age"]
2. df[["name", "age"]]
```

Copied!

Syntax:

```
1. 1
```

```
1. dataframe_name.describe()
```

Copied!

describe()

Generates statistics summary of numeric columns in the DataFrame.

Example:

```
1. 1
```

```
1. df.describe()
```

Copied!

	Syntax:
	<pre> 1. 1 2. 2 1. dataframe_name.drop(["column1", "column2"], axis=1, inplace=True) 2. dataframe_name.drop(index=[row1, row2], axis=0, inplace=True) </pre>
drop()	<p>Removes specified rows or columns from the DataFrame. axis=1 indicates columns. axis=0 indicates rows.</p>
	<p style="text-align: right;">Copied!</p>
	Example:
	<pre> 1. 1 2. 2 1. df.drop(["age", "salary"], axis=1, inplace=True) # Will drop columns 2. df.drop(index=[5, 10], axis=0, inplace=True) # Will drop rows </pre>
	<p style="text-align: right;">Copied!</p>
	Syntax:
	<pre> 1. 1 1. dataframe_name.dropna(axis=0, inplace=True) </pre>
dropna()	<p>Removes rows with missing NaN values from the DataFrame. axis=0 indicates rows.</p>
	<p style="text-align: right;">Copied!</p>
	Example:
	<pre> 1. 1 1. df.dropna(axis=0, inplace=True) </pre>
	<p style="text-align: right;">Copied!</p>
	Syntax:
	<pre> 1. 1 1. dataframe_name.duplicated() </pre>
duplicated()	<p>Duplicate or repetitive values or records within a data set.</p>
	<p style="text-align: right;">Copied!</p>
	Example:
	<pre> 1. 1 1. duplicate_rows = df[df.duplicated()] </pre>
	<p style="text-align: right;">Copied!</p>
	Syntax:
	<pre> 1. 1 1. filtered_df = dataframe_name[(Conditional_statements)] </pre>
Filter Rows	<p>Creates a new DataFrame with rows that meet specified conditions.</p>
	<p style="text-align: right;">Copied!</p>
	Example:
	<pre> 1. 1 1. filtered_df = df[(df["age"] > 30) & (df["salary"] < 50000) </pre>
	<p style="text-align: right;">Copied!</p>
	Syntax:
	<pre> 1. 1 2. 2 1. grouped = dataframe_name.groupby(by, axis=0, level=None, as_index=True, 2. sort=True, group_keys=True, squeeze=False, observed=False, dropna=True) </pre>
groupby()	<p>Splits a DataFrame into groups based on specified criteria, enabling subsequent aggregation, transformation, or analysis within each group.</p>
	<p style="text-align: right;">Copied!</p>
	Example:
	<pre> 1. 1 1. grouped = df.groupby(["category", "region"]).agg({"sales": "sum"}) </pre>
	<p style="text-align: right;">Copied!</p>
	Syntax:
	<pre> 1. 1 1. dataframe_name.head(n) </pre>
head()	<p>Displays the first n rows of the DataFrame.</p>
	<p style="text-align: right;">Copied!</p>
	Example:
	<pre> 1. 1 </pre>

		1. df.head(5)
		Copied!
		Syntax:
		1. 1
		1. import pandas as pd
		Copied!
Import pandas	Imports the Pandas library with the alias pd.	Example:
		1. 1
		1. import pandas as pd
		Copied!
		Syntax:
		1. 1
		1. dataframe_name.info()
		Copied!
info()	Provides information about the DataFrame, including data types and memory usage.	Example:
		1. 1
		1. df.info()
		Copied!
		Syntax:
		1. 1
		1. merged_df = pd.merge(df1, df2, on=["column1", "column2"])
		Copied!
merge()	Merges two DataFrames based on multiple common columns.	Example:
		1. 1
		1. merged_df = pd.merge(sales, products, on=["product_id", "category_id"])
		Copied!
		Syntax:
		1. 1
		1. print(df) # or just type df
		Copied!
print DataFrame	Displays the content of the DataFrame.	Example:
		1. 1
		2. 2
		1. print(df)
		2. df
		Copied!
		Syntax:
		1. 1
		1. dataframe_name["column_name"].replace(old_value, new_value, inplace=True)
		Copied!
replace()	Replaces specific values in a column with new values.	Example:
		1. 1
		1. df["status"].replace("In Progress", "Active", inplace=True)
		Copied!
		Syntax:
		1. 1
		1. dataframe_name.tail(n)
		Copied!
tail()	Displays the last n rows of the DataFrame.	Example:
		1. 1

```
1. df.tail(5)
```

Copied!

Numpy

Package/Method	Description	Syntax and Code Example
Importing NumPy	Imports the NumPy library.	<p>Syntax:</p> <pre>1. 1</pre> <p>1. import numpy as np</p> <p>Copied!</p>
np.array()	Creates a one or multi-dimensional array,	<p>Example:</p> <pre>1. 1</pre> <pre>1. import numpy as np</pre> <p>Copied!</p> <p>Syntax:</p> <pre>1. 1</pre> <pre>2. 2</pre> <pre>1. array_1d = np.array([list1 values]) # 1D Array</pre> <pre>2. array_2d = np.array([[list1 values], [list2 values]]) # 2D Array</pre> <p>Copied!</p> <p>Example:</p> <pre>1. 1</pre> <pre>2. 2</pre> <pre>1. array_1d = np.array([1, 2, 3]) # 1D Array</pre> <pre>2. array_2d = np.array([[1, 2], [3, 4]]) # 2D Array</pre> <p>Copied!</p>
Numpy Array Attributes	- Finds the minimum value in the array - Finds the maximum value in the array - Computes dot product of two arrays	<ul style="list-style-type: none">- Calculates the mean of array elements- Calculates the sum of array elements <pre>1. np.mean(array)</pre> <pre>2. np.sum(array)</pre> <pre>3. np.min(array)</pre> <pre>4. np.max(array)</pre> <pre>5. np.dot(array_1, array_2)</pre>

Copied!



Skills Network