

# 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
Iterating over lines	Iterates through each line in the file using a 'loop'.	Copied!  Example:  1. 1 2. 2  1. with open("data.txt", "r") as file: 2. for line in file: print(line)
		Copied!  Syntax:  1. 1 2. 2  1. file = open(filename, mode) # Code that uses the file 2. file.close()
Open() and close()	Opens a file, performs operations, and explicitly closes the file using the close() method.	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, ensuring automatic file closure after usage.

Copied!

Example:

```
1. 1
2. 2
```

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

Copied!

with open()

Pandas

Package/Method	Description	Syntax and Code Example
.read_csv()	Reads data from a '.CSV' file and creates a DataFrame.	<div>Syntax: dataframe_name = pd.read_csv("filename.csv") Example: df = pd.read_csv("data.csv")</div> <div>Syntax:</div> <div>1. 1</div> <div>1. dataframe_name = pd.read_excel("filename.xlsx")</div> <div>Copied!</div>
.read_excel()	Reads data from an Excel file and creates a DataFrame.	<div>Example:</div> <div>1. 1</div> <div>1. df = pd.read_excel("data.xlsx")</div> <div>Copied!</div> <div>Syntax:</div> <div>1. 1</div> <div>1. dataframe_name.to_csv("output.csv", index=False)</div> <div>Copied!</div>
.to_csv()	Writes DataFrame to a CSV file.	<div>Example:</div> <div>1. 1</div> <div>1. df.to_csv("output.csv", index=False)</div> <div>Copied!</div> <div>Syntax:</div> <div>1. 1</div> <div>2. 2</div> <div>1. dataframe_name["column_name"] # Accesses single column</div> <div>2. dataframe_name[["column1", "column2"]] # Accesses multiple columns</div> <div>Copied!</div>
Access Columns	Accesses a specific column using [] in the DataFrame.	<div>Example:</div> <div>1. 1</div> <div>2. 2</div> <div>1. df["age"]</div> <div>2. df[["name", "age"]]</div> <div>Copied!</div> <div>Syntax:</div> <div>1. 1</div> <div>1. dataframe_name.describe()</div> <div>Copied!</div>
describe()	Generates statistics summary of numeric columns in the DataFrame.	<div>Example:</div> <div>1. 1</div> <div>1. df.describe()</div> <div>Copied!</div>

Syntax:

```
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)
```

Copied!

Example:

```
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
```

Copied!

Syntax:

```
1. 1
```

```
1. dataframe_name.dropna(axis=0, inplace=True)
```

Copied!

Example:

```
1. 1
```

```
1. df.dropna(axis=0, inplace=True)
```

Copied!

Syntax:

```
1. 1
```

```
1. dataframe_name.duplicated()
```

Copied!

Example:

```
1. 1
```

```
1. duplicate_rows = df[df.duplicated()]
```

Copied!

Syntax:

```
1. 1
```

```
1. filtered_df = dataframe_name[(Conditional_statements)]
```

Copied!

Example:

```
1. 1
```

```
1. filtered_df = df[(df["age"] > 30) & (df["salary"] < 50000)]
```

Copied!

Syntax:

```
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)
```

Copied!

Example:

```
1. 1
```

```
1. grouped = df.groupby(["category", "region"]).agg({"sales": "sum"})
```

Copied!

Syntax:

```
1. 1
```

```
1. dataframe_name.head(n)
```

Copied!

Example:

```
1. 1
```

drop()

Removes specified rows or columns from the DataFrame. axis=1 indicates columns. axis=0 indicates rows.

dropna()

Removes rows with missing NaN values from the DataFrame. axis=0 indicates rows.

duplicated()

Duplicate or repetitive values or records within a data set.

Filter Rows

Creates a new DataFrame with rows that meet specified conditions.

groupby()

Splits a DataFrame into groups based on specified criteria, enabling subsequent aggregation, transformation, or analysis within each group.

head()

Displays the first n rows of the DataFrame.

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

```
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> <pre>1. import numpy as np</pre> <p>Example:</p> <pre>1. 1</pre> <pre>1. import numpy as np</pre>
np.array()	Creates a one or multi-dimensional array,	<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>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>
Numpy Array Attributes	<ul style="list-style-type: none"><li>- Calculates the mean of array elements</li><li>- Calculates the sum of array elements</li><li>- Finds the minimum value in the array</li><li>- Finds the maximum value in the array</li><li>- Computes dot product of two arrays</li></ul>	<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