On this page you will find a collection of useful pdf files and code snippets.

Overview of Important Python Syntax

Data Types	Operators	Control Structures	Loops	Libraries
Integers	Addition (+)	If Statements	For Loop	numpy
`x = 5`	`result = a + b`	`if x > 5:`	`for i in range(10):`	`import numpy as np`
Floats	Subtraction (-)	Else Statements	While Loop	pandas
`y = 3.5`	`result = a - b`	`else:`	`while x > 0:`	`import pandas as pd`
Strings	Multiplication (*)	Elif Statements		matplotlib
`name = "John"`	`result = a * b`	`elif x < 10:`		`import matplotlib.pyplot as plt`
Lists	Division (/)	Try and Catch		
`my_list = [1, 2, 3]`	`result = a / b`	`try:`		
Data Frames	Modulus (%)	Break and Continue		
`df = pd.DataFrame(data)`	`result = a % b`	`break # Exit the loop when`		
Arrays	Exponentiation (**)	`continue # Skip the even numbers`		
`my_array = np.array([1, 2, 3])`	`result = a ** b`			
Matrix	Boolean			
`my_matrix = np.array([[1, 2], [3, 4]])`	`<, >, ==, >=, <=, !=`			

Basics of Syntax

Python is known for its simple and readable syntax. Here are some basic rules:

- No semicolon (;) at the end of a line.
- Indentation with 4 spaces instead of curly braces ({}) for code blocks.
- Comments start with a #.

```
# This is a comment
print("Hello, World!")  # Outputs Hello, World!
```

Data Types and Variables

Python is dynamically typed. Variable assignment is done simply with the = sign.

Examples of Data Types:

- int (Integers)
- float (Floating-point numbers)
- str (Strings)
- bool (Boolean)

```
x = 10  # int
y = 3.14  # float
name = "Alice"  # str
is_student = True  # bool
```

Control Structures (If, Loops)

If-Else

```
if x > 5:
    print("x is greater than 5")
else:
    print("x is 5 or smaller")
```

For Loop

```
for i in range(5):
    print(i)  # Outputs 0 to 4
```

While Loop

```
n = 0
while n < 5:
```

```
    print(n)
    n += 1
```

Error Handling

You can catch errors with try and except.

```
try:
    result = 10 / 0
except ZeroDivisionError:
    print("Division by zero is not allowed")
```

Functions and Methods

Functions in Python are defined with the def keyword.

def greet(name): return f"Hello, {name}!"

```
print(greet("Bob"))  # Outputs "Hello, Bob!"
```

Objects

Lists,

```
fruits = ["Apple", "Banana", "Cherry"]
print(fruits[1])  # Outputs "Banana"
```

Conclusion

Lists are flexible data structures in Python that can contain elements of different data types. Arrays, on the other hand, are homogeneous data structures that store elements of the same data type. NumPy arrays provide superior performance for mathematical operations compared to Python lists and are optimized for scientific computing.

Data Frames

```
import pandas as pd

# Creating lists
a = ["Max", "Sara"]
b = [24, 42,]

# Creating a data frame from lists with assigned column names
```

```
patients = pd.DataFrame({
    'Name': a,
    'Age': b,
})
```

Matrix

```
import numpy as np
# Create a matrix
M = np.array([[3,5,6], [11,76,4], [0,7,99]])
```

Conclusion

DataFrames are versatile data structures provided by the Pandas library in Python. They allow for the storage and manipulation of tabular data with labeled axes (rows and columns). DataFrames are particularly useful for data analysis and manipulation.

Matrices, on the other hand, are primarily used for numerical computations. While matrices are suited for mathematical operations such as matrix multiplication and linear algebra, they lack the rich functionalities of DataFrames for data manipulation and analysis.

Reading and Writing Files

Reading

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

Writing

```
with open("file.txt", "w") as file:
    file.write("Hello, World!")
```

Modules and Packages

Modules in Python are collections of functions. You can use them with import.

```
import math

print(math.sqrt(16))  # Outputs 4.0
```

Useful Libraries

NumPy

NumPy is a library for numerical computations.

```
import numpy as np

a = np.array([1, 2, 3])
print(a * 2)  # Outputs [2, 4, 6]
```

Pandas

Pandas is great for data analysis.

```
import pandas as pd

data = {'Name': ['Alice', 'Bob'], 'Age': [25, 30]}
df = pd.DataFrame(data)
print(df)
```