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 Except		
`my_list = [1, 2, 3]`	`result = a / b`	`try:`		
DataFrames	Modulus (%)	Break and Continue		
`df = pd.DataFrame(data)`	`result = a % b`	`break` / `continue`		
Arrays	Exponentiation (**)			
`np.array([1, 2, 3])`	`result = a ** b`			

Basic Data Types

Python is dynamically typed, meaning variables do not need explicit declarations. Common basic data types:

Integers, Floats, Strings, Booleans

```
x = 10          # Integer
y = 3.14        # Float
name = "Alice"   # String
is_student = True  # Boolean
```

Object Data Types

Arrays

Arrays come from the NumPy library and allow efficient operations on large data sets. Arrays are homogeneous (one data type).

```
import numpy as np

arr = np.array([1, 2, 3, 4])
print(arr * 2)  # Outputs [2, 4, 6, 8]
```

DataFrames

DataFrames are powerful table-like data structures from Pandas, great for analysis.

```
import pandas as pd

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

Importing and Exporting Data with Pandas

Read data

```
import pandas as pd

df = pd.read_csv("data.csv")
df_excel = pd.read_excel("data.xlsx")
```

Write data

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

Visualization with Matplotlib

Matplotlib allows you to create plots and visualizations easily.

```python
import matplotlib.pyplot as plt

x = [1, 2, 3, 4]
y = [10, 20, 25, 30]

plt.plot(x, y, label="Line")
plt.xlabel("X-axis")
plt.ylabel("Y-axis")
plt.title("Simple Plot")
plt.legend()
plt.show()
```

Control Structures

If-Else

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

For Loop

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

While Loop

```python
n = 0
while n < 5:
    print(n)
    n += 1
```

Error Handling (Try and Except)

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

Object-Oriented Programming (OOP)

OOP is a paradigm that organizes code using objects, which bundle data (attributes) and behavior (methods).

```
class Vehicle:
    def __init__(self, brand, year):
        self.brand = brand
        self.year = year

    def display_info(self):
        print(f"Brand: {self.brand}, Year: {self.year}")

# Create an object
car = Vehicle("Toyota", 2020)
car.display_info()
```

Modules and Packages

Modules are Python files containing functions, classes, and variables. You can import them using import.

```
import math

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

Useful Libraries

NumPy: Fast numerical operations on arrays and matrices.

Pandas: Data analysis and manipulation with DataFrames.

Matplotlib: Creating plots and visualizations.