00_PythonReviewGuide

Python Programming Review Guide

This guide provides a concise review of essential Python programming concepts, ideal for exam preparation or refreshing your knowledge.

Table of Contents

Introduction

Python is a powerful, high-level programming language known for its simplicity and readability, making it ideal for beginners and experienced developers alike.

Python Basics

Syntax and Variables

- **Variables**: Automatically assigned data types, created upon assignment.
- **Comments**: Use `#` for single-line comments and `'''` or `"""` for multi-line comments.

Basic Data Types

- **Integers, Floats, Strings, Booleans**.
- **Conversion**: Use `int()`, `float()`, `str()`, and `bool()` for type conversion.

Operators

- **Arithmetic**: `+`, `-`, `*`, `/`, `//`, `%`, `**`
- **Comparison**: `==`, `!=`, `>`, `<`, `>=`, `<=`
- **Logical**: `and`, `or`, `not`
- **Membership**: `in`, `not in`

Data Structures

Lists

- **Definition**: Ordered, mutable collections.
- **Common Methods**: `append()`, `remove()`, `pop()`, `reverse()`, `sort()`

Tuples

- **Definition**: Ordered, immutable collections.
- **Usage**: Useful for fixed data, faster than lists.

Dictionaries

- **Definition**: Key-value pairs, unordered, mutable.
- **Accessing**: `dict[key]`, `dict.get(key)`
- **Methods**: `keys()`, `values()`, `items()`

Sets

- **Definition**: Unordered collections of unique elements.
- **Operations**: `add()`, `remove()`, `union()`, `intersection()`

Control Structures

If Statements

- **Syntax**:

```
if condition:
    # do something
elif another_condition:
    # do something else
else:
    # do another thing
```

Loops

- **For Loops**: Iterating over a sequence.
- **While Loops**: Repeating as long as a condition is true.

Functions

- **Definition**: `def function_name(parameters):`
- **Return Values**: Use `return` to return values.
- **Lambda Functions**: Anonymous functions, defined with `lambda` keyword.

Modules and Packages

- **Importing**: `import module_name` or `from module import function_name`
- **Creating Modules**: Save your functions in a `.py` file and import them.

File Handling

- **Reading and Writing Files**:

```
with open('filename.txt', 'r') as file:
    content = file.read()
```

Exception Handling

- **Try and Except**:

```
try:
    # try to execute code
except ErrorType:
    # handle the exception
finally:
    # execute code regardless of the exception
```

Libraries: NumPy and Matplotlib

NumPy

- **Arrays**: Core feature, used for numerical operations.
- **Functions**: `np.array()`, `np.zeros()`, `np.ones()`, `np.arange()`

Matplotlib

- **Plotting**: `plt.plot()`, `plt.show()`
- **Subplots**: `plt.subplot()`