Python Cheatsheet

Python is a versatile programming language used for various applications.

---

1. Basic Syntax and Data Types

Printing and Comments

- **print("Hello, World!")**: Prints output to the console.
- **# This is a comment**: Single-line comment.
- **'''This is a multi-line comment'''**: Multi-line comment.

Variables and Data Types

- **x = 10:** Assigns a value to a variable.
- **type(x):** Returns the data type of a variable.
- **a, b = 5, 10:** Multiple assignments.
- **del x:** Deletes a variable.
- **int(), float(), str(), bool()**: Type conversions.
- **isinstance(x, int):** Checks data type.
- **complex(2, 3)**: Creates a complex number.
- **bin(10), hex(255), oct(8)**: Converts numbers to binary, hexadecimal, and octal.
- **id(x):** Returns memory address of a variable.

---

2. Data Structures

Lists (Ordered, Mutable)

- **lst = [1, 2, 3]:** Creates a list.
- **lst.append(4)**: Adds an element.
- **lst.pop():** Removes and returns the last element.
- **lst.sort():** Sorts the list.
- **lst.reverse():** Reverses the list.
- **lst.insert(1, 10):** Inserts an element at an index.
- **lst.remove(2):** Removes the first occurrence of an element.
- **lst.count(3):** Counts occurrences of an element.
- **lst.index(3):** Finds index of an element.

- **len(lst), max(lst), min(lst), sum(lst):** Common list operations.

Tuples (Ordered, Immutable)

- **tup = (1, 2, 3):** Creates a tuple.
- **tup.count(1):** Counts occurrences of an element.
- **tup.index(2):** Finds index of an element.
- **tuple(lst):** Converts list to tuple.

Dictionaries (Key-Value Pairs, Unordered)

- **d = {'a': 1, 'b': 2}:** Creates a dictionary.
- **d['a']:** Accesses a value.
- **d.keys():** Returns all keys.
- **d.values():** Returns all values.
- **d.items():** Returns key-value pairs.
- **d.get('a', 0):** Gets a value with a default.
- **d.update({'c': 3}):** Updates the dictionary.
- **d.pop('b'):** Removes a key.
- **dict.fromkeys(['a', 'b'], 0):** Creates dictionary with default values.

Sets (Unordered, Unique Items)

- **s = {1, 2, 3}:** Creates a set.
- **s.add(4):** Adds an element.
- **s.remove(2):** Removes an element.
- **s.union({5, 6}):** Combines sets.
- **s.intersection({1, 2}):** Finds common elements.
- **s.difference({1}):** Finds unique elements in s.

---

3. Control Flow

Conditional Statements

- **if x > 0: print("Positive"):** If statement.
- **elif x == 0: print("Zero"):** Else-if statement.
- **else: print("Negative"):** Else statement.

Loops

- **for i in range(5):** print(i) - For loop.

- **while x > 0: x -= 1:** While loop.
- **break, continue, pass:** Loop control statements.
- **enumerate(lst):** Iterates with index.
- **zip(lst1, lst2):** Iterates over two lists.

---

4. Functions and Modules

- **def func(a, b):** return a + b - Defines a function.
- **lambda x: x * 2:** Anonymous function.
- **import math:** Imports a module.
- **from math import sqrt:** Imports a specific function.
- **dir(math):** Lists module attributes.
- **globals(), locals():** Access global and local variables.

---

5. Object-Oriented Programming

- **class Person:** Defines a class.
- **def __init__(self, name):** Constructor.
- **self.name = name**: Instance variable.
- **p = Person("Alice"):** Object instantiation.
- **class Employee(Person):** Inheritance.
- **super().__init__()**: Calls parent constructor.
- **@staticmethod, @classmethod**: Defines static and class methods.

---

6. Exception Handling

- **try: x = 1 / 0 except ZeroDivisionError:** print("Error") - Exception handling.
- **finally: print("Done"):** Executes always.
- **raise ValueError("Invalid"):** Manually raises an exception.

---

7. File Handling

- **with open('file.txt', 'r') as f: data = f.read():** Reads a file.
- **f.write("Hello"):** Writes to a file.
- **f.close():** Closes the file.

- **open('file.txt', 'w'):** Opens file in write mode.

---

8. Advanced Topics

List Comprehensions

- **[x**2 for x in range(10)]:** Creates a list with squares.
- **{x: x**2 for x in range(5)}:** Dictionary comprehension.

Decorators

- **@decorator def func(): pass** - Defines a decorator.
- **@property** - Creates a read-only attribute.

Generators

- **def gen(): yield 1 - Defines a generator.**
- **next(gen())** - Retrieves the next value from a generator.

Threading and Multiprocessing

- **import threading** - Multi-threading support.
- **import multiprocessing** - Parallel processing support.

Regular Expressions

- **import re** - Regex support.
- **re.match(pattern, string)** - Matches pattern.
- **re.findall(r'\d+', text)** - Extracts numbers.

JSON Handling

- **import json** - JSON support.
- **json.dumps(data)** - Converts Python object to JSON.
- **json.loads(json_string)** - Converts JSON to Python object.

Official documentation: https://docs.python.org/3/