Python Theory Questions and Answers

# Basic Python Topics

## What are the different data types available in Python? How is a variable declared in Python?

Python supports data types such as int, float, str, bool, list, tuple, set, and dict. Variables are declared by simply assigning a value: x = 5, name = 'John'.

## What are the different types of operators in Python? Explain with examples.

Python includes arithmetic (+, -, \*), comparison (==, !=), logical (and, or), assignment (=, +=), bitwise, membership (in, not in), and identity (is, is not) operators.

## How does the if-elif-else structure work in Python? Give an example.

It evaluates conditions in order. If one is True, it runs that block. Example:  
if x > 0:  
 print('Positive')  
elif x == 0:  
 print('Zero')  
else:  
 print('Negative')

## Differentiate between for and while loops in Python. When would you use each?

'for' loops iterate over sequences; 'while' loops run as long as a condition is True. Use 'for' when you know the number of iterations.

## What is a function in Python? How do you define and call a function?

A function is a block of reusable code. Define using 'def':  
def greet():  
 print('Hello')  
Call with: greet()

## What are the differences between lists, tuples, sets, and dictionaries in Python?

Lists: ordered, mutable; Tuples: ordered, immutable; Sets: unordered, unique; Dictionaries: key-value pairs.

## How can you manipulate strings in Python? List some common string methods.

Use methods like upper(), lower(), replace(), split(), join(), strip(). Example: 'hello'.upper() returns 'HELLO'.

## How does Python handle user input and output?

Use input() to get user input, print() to display output. Example: name = input('Enter name: ')

## What is exception handling in Python? How is it implemented using try-except?

Use try-except blocks to catch errors. Example:  
try:  
 x = 1/0  
except ZeroDivisionError:  
 print('Cannot divide by zero')

# Intermediate Python Topics

## How do you read and write files in Python? Explain the file modes.

Use open(). Modes: 'r' (read), 'w' (write), 'a' (append), 'rb' (read binary). Example:  
f = open('file.txt', 'r')  
data = f.read()  
f.close()

## What is list comprehension? Give an example and its advantages.

It's a concise way to create lists. Example: [x for x in range(5) if x % 2 == 0] => [0, 2, 4]

## What is a lambda function in Python? How does it differ from a regular function?

Lambda is an anonymous, single-expression function. Example: square = lambda x: x\*x

## What is the difference between a module and a package in Python? How do you import them?

A module is a single .py file. A package is a directory with \_\_init\_\_.py. Use import module or from package import module.

## Explain the four principles of OOP in Python with examples.

Encapsulation, Abstraction, Inheritance, Polymorphism. Example: A class hides data (encapsulation) and inherits from another class (inheritance).

## What is a decorator in Python? How and why would you use one?

A decorator modifies function behavior. Use @decorator\_name above a function. Example: logging or authentication.

## Differentiate between iterators and generators in Python. Provide an example.

Iterators use \_\_iter\_\_() and \_\_next\_\_(). Generators use yield and are more memory efficient. Example:  
def gen():  
 yield 1

# Advanced Python Topics

## What are regular expressions in Python and how are they used?

Used for string matching. Import re module. Example: re.search('pattern', text)

## What is the difference between multithreading and multiprocessing in Python?

Multithreading shares memory space; multiprocessing uses separate processes. Multiprocessing is better for CPU-bound tasks.

## What is a context manager in Python? How does the 'with' statement work?

Context managers manage resources. 'with' auto handles setup/teardown. Example: with open('file.txt') as f

## Briefly explain the use of libraries like NumPy, Pandas, and Matplotlib.

NumPy: arrays and math; Pandas: data analysis; Matplotlib: plotting and visualization.

## What is a virtual environment in Python and why is it important?

It's an isolated environment for dependencies. Prevents conflicts between projects. Create using: python -m venv env

## What are Flask and Django? How do they differ and when would you use each?

Flask: lightweight, flexible. Django: full-featured, batteries-included. Use Flask for small apps, Django for larger ones.

## How can you make API calls in Python? Explain the use of the requests library.

Use requests.get(), requests.post(), etc. Example:  
import requests  
r = requests.get('https://api.example.com')

## How do you perform unit testing in Python? What are unittest and pytest?

Use unittest or pytest to write test cases. unittest is built-in; pytest is simpler and more powerful for advanced tests.