**PYTHON**

1. Basic Python Syntax

Understanding Python's syntax, data types, variables, and basic operators.

2. Control Structures

Using if, else, elif, for, and while loops for conditional logic and iteration.

arrays,strings,list,sets,tuples,dictinaries

3. Functions

Defining and calling functions, understanding scope, and using lambda functions.

4. Modules and Packages

Importing and creating modules, understanding Python packages, and using pip for package management.

5. Exception Handling

Using try, except, finally, and raise for managing errors and exceptions.

6. File I/O

Reading from and writing to files, handling file paths, and working with different file formats (e.g., text, CSV).

7. Working with JSON and YAML

Parsing and generating JSON and YAML files for configuration management.

8. Regular Expressions

Using the re module for pattern matching and text manipulation.

9. Networking with Sockets

Using the socket module for basic networking operations.

10. HTTP Requests

Using the requests library to interact with web services and APIs.

11. Automation and Scheduling

Automating tasks with Python scripts and scheduling them using tools like cron or APScheduler.

12. SSH and SFTP

Automating remote server management with libraries like paramiko and pysftp.

13. Web Scraping

Using libraries like BeautifulSoup and Scrapy to extract data from web pages.

14. Working with APIs

Interacting with RESTful APIs and handling authentication, requests, and responses.

Building and Consuming Web Services

Using Flask, Django, or FastAPI for creating web services.

15. Docker and Containerization

Using the docker library to manage Docker containers and images programmatically.

16. Cloud Services Automation

Automating tasks in cloud environments using SDKs such as boto3 for AWS, google-cloud for GCP, and azure-mgmt for Azure.

17. Infrastructure as Code (IaC)

Writing scripts for infrastructure automation using tools like AWS CDK, Terraform, and Ansible.

18. Testing and Validation

Writing unit tests and integration tests with frameworks like pytest and unittest.

19. Logging and Monitoring

Implementing logging using the logging module and monitoring systems using tools like Prometheus and Grafana.

20. Continuous Integration and Deployment (CI/CD)

Integrating Python scripts with CI/CD pipelines using tools like Jenkins, GitLab CI, and GitHub Actions.

21. Data Serialization

Serializing and deserializing data using formats like JSON, XML, and binary formats.

22. Working with Databases

Interacting with relational databases using sqlalchemy and sqlite3, and NoSQL databases using libraries like pymongo for MongoDB.

23. System Administration

Performing system tasks such as managing users, processes, and system resources with Python.

24. Environment Management

Using virtual environments with venv or virtualenv to manage project dependencies.

25. Performance Optimization

Profiling and optimizing Python code for better performance and resource usage.

26. Security Practices

Implementing secure coding practices and handling sensitive data safely.

27. Network Automation

Using libraries like netmiko and nornir to automate network device management.

28. Configuration Management

Managing configuration files and settings programmatically.

29. Version Control Integration

Automating interactions with version control systems like Git using gitpython or command-line tools.

30. Automation Frameworks

Using frameworks like Fabric and Invoke for task automation and orchestration.

31. Data Analysis

Using libraries like pandas and numpy for analyzing and manipulating data.

32. JSON Schema Validation

Validating JSON data against schemas to ensure data integrity.

33. Dependency Management

Managing project dependencies and versions using pip, pipenv, or poetry.

34. CLI Development

Building command-line interfaces with libraries like argparse or click.

35. Virtualization and Container Orchestration

Understanding container orchestration tools like Kubernetes and their integration with Python.

36. Encryption and Decryption

Implementing cryptographic operations using libraries like cryptography and pycryptodome.

37. Time and Date Manipulation

Working with date and time using datetime and dateutil.

38. Parallel and Concurrent Programming

Utilizing threading, multiprocessing, and asyncio for concurrent tasks.

39. Networking Protocols

Understanding and implementing various networking protocols and services (e.g., HTTP, FTP, DNS).

40. Documentation and Code Quality

Writing clear documentation and adhering to code quality standards using tools like Sphinx and flake8.

41.Memory Management

Garbage Collection: Understanding and managing Python's memory usage.

42.Data Manipulation with pandas

DataFrames: Handling and analyzing large datasets with pandas

43.Integrating with Message Brokers

Using pika for RabbitMQ or kafka-python for Kafka: Handling message queues.

44.Handling Large Files

Processing large files efficiently: Using streaming and chunking techniques.

45.Using Environment-Specific Configurations

Managing different configurations for development, testing, and production.

46.Deploying Python Applications

Packaging and deploying Python applications: Using tools like gunicorn for web apps or pyinstaller for standalone executables.

47.Advanced Debugging Techniques

Using tools and techniques for debugging complex Python applications.