Python 3.11 Essentials

- Syntax & types
- Control flow
- Functions & modules
- Exceptions & context managers

Setup & REPL

- Create venv and activate
- `python --version` should show 3.11+
- Use `python -q` for quick checks

Types & Literals

- str, int, float, bool, None
- list, dict, tuple, set
- f-strings for formatting

```
name = "interface"
num = 123
print(f"{name}-{num}")
```

Control Flow

- if / elif / else
- for / while
- comprehension patterns

```
up_ifaces = [i for i in ifaces if i["status"] == "up"]
```

Functions & Modules

- def, return, default args, type hints
- `import requests as r` (aliasing)
- `from common.lib import inventory`

```python
def parse(text: str) -> dict:
    return {"lines": len(text.splitlines())}
```

Exceptions

- try / except / else / finally
- raise vs return error codes
- Wrap network I/O with timeouts + retries later

```python
try:
    risky()
except TimeoutError as e:
    log.warning("timeout: %s", e)
```

With-Statements (Context Managers)

- Auto-close connections/files
- `with ConnectHandler(**params) as conn:`

```python
with open("output.json", "w") as f:
    f.write("{}")
```

Logging Basics

- INFO to file; warnings to console
- Use `common.lib.logging_setup.setup_logging(name)`

JSON & Dicts

- `json.loads()` / `json.dumps(..., indent=2)`
- Dict access: `.get()` with defaults

CLI Args & Argparse (Preview)

- `argparse.ArgumentParser()`
- `--apply` vs `--dry-run`

Practice

- Convert a list of CLI lines to JSON
- Count interfaces by state
- Handle missing keys safely