Session 1.2: Essential Python Syntax Completion, Control Flow, Collections, and Business Automation

TODAY'S PURPOSE

Why This Session Matters

- Turn written business rules into Python logic
- Let code decide and repeat automatically
- Stop copying formulas row by row

Today's flow: 60 min lecture, then 120 min hands-on lab

EXPECTED OUTCOMES

By the end of today, you will be able to:

1. Apply tiered rules with if / elif / else
2. Process client records with lists, dictionaries, and loops
3. Fix an IndentationError in a full script

Five Concepts That Make Business Rules Executable

01

Boolean Expression

A comparison that becomes `True` or `False`, such as a spend threshold.

Conditional Branching

`if`, `elif`, and `else` select the right business action.

02

List Collection

An ordered, mutable sequence of records processed from first item to last.

Dictionary Collection

A labeled record that retrieves values by useful names, such as `"annual_spend_USD"`.

03

Definite Iteration

A `for` loop applies the same policy to every record with consistent logic and speed.

Readable Rules Are Auditable Rules

FROM SPREADSHEETS

Replace Nested Formulas

Long nested spreadsheet formulas are difficult to audit. Python uses visible vertical branches that mirror a manager's decision tree.

TO PROGRAMS

Use Whitespace Deliberately

Python indentation defines the scope of a branch or loop. Four aligned spaces make logic readable and prevent accidental rule changes.

AT SCALE

Repeat Without Fatigue

A loop applies one policy to ten or one million records with the same calculation every time.

WITH CONTEXT

Model Real Entities

Dictionaries preserve customer attributes such as industry, spend, credit limit, and risk score in one record.

Use the IDE to Build, Test, and Diagnose

[1] FILE EXPLORER

Organize Inputs

Store the session dataset in `data/session_01_02_dataset.csv`.

[2] CODE EDITOR

Author the Script

Write one small, executable segment at a time.

[3] AI AGENT CHAT

Explain the Error

Paste a traceback and ask for the cause, the correction, and how to verify it.

[4] INTEGRATED TERMINAL

Confirm the Result

Run the script and verify return code 0 after every correction.

Capture protocol: Identify the final error line in Terminal, capture it with Windows Key + Shift + S or Command + Shift + 4, paste it into AI Chat, apply the verified fix, and run again.

Script Demonstration 1: Conditional Customer Tiering

session_01_02_demo_1.py | Python 3.10

```
annual_spend = 82000.0

if annual_spend >= 100000.0:
    tier = "Platinum"
    discount_rate = 0.15
elif annual_spend >= 50000.0:
    tier = "Gold"
    discount_rate = 0.10
else:
    tier = "Standard"
    discount_rate = 0.0

print(f"Customer Tier: {tier}")
```

```
$ python session_01_02_demo_1.py [Exit 0]
```

SYNTAX

Aligned Branches

`if`, `elif`, and `else` begin at the same indentation level.

RUNTIME

One Rule Applies

Only the first matching branch runs, assigning one tier and discount rate.

BUSINESS VALUE

Auditable Policy

The rule can be reviewed line by line and changed without rewriting a nested formula.

Script Demonstration 2: Iterative Revenue Accumulation

session_01_02_demo_2.py | Python 3.10

```
daily_sales = [4200.0, 5100.0, 3900.0, 6200.0]
total_revenue = 0.0

for sale in daily_sales:
    total_revenue += sale

average_sales = total_revenue / len(daily_sales)
print("Total Revenue:", total_revenue)
print("Average Daily Sales:", average_sales)
```

```
$ python session_01_02_demo_2.py [Exit 0]
```

SYNTAX

Loop Header

The colon opens a block; the indented statement executes once for each sale.

RUNTIME

Accumulator State

`total_revenue` retains its updated value as the loop advances.

BUSINESS VALUE

Consistent Aggregation

The same calculation works whether the list has four transactions or four million.

Script Demonstration 3: Dictionary-Based Credit Review

session_01_02_demo_3.py | Python 3.10

```
client = {
    "company_name": "Apex Logistics",
    "credit_limit_USD": 75000.0,
    "outstanding_balance_USD": 23400.0,
}

available_credit = (
    client["credit_limit_USD"]
    - client["outstanding_balance_USD"]
)
print(client["company_name"], available_credit)
```

```
$ python session_01_02_demo_3.py [Exit 0]
```

SYNTAX

Named Attributes

Dictionary keys make the meaning of each number explicit.

RUNTIME

Safe Calculation

The calculation retrieves two labeled values before subtracting them.

BUSINESS VALUE

Useful Record Model

One customer object keeps related operational information together.

Lab: Automate a Client Portfolio

EXERCISE 1: GUIDED

Client Portfolio Automation

Load the session dataset, build customer dictionaries, assign a tier using spend and risk, and print each approved credit limit.

Success criterion: every client receives one tier and the script finishes with return code 0.

EXERCISE 2: DEBUGGING

Commission Pipeline

Write a loop for three sales representatives. Intentionally misalign one line, capture the resulting `IndentationError`, obtain an explanation, correct the code, and verify clean execution.

Deliverable: a formatted compensation summary with commission payouts.