Chapter 5: Logical and Assignment Operators

How Python makes decisions and remembers changes.

🧠 Introduction

In real life, we make decisions:

- “If I finish my work **and** it's sunny, I'll go out.”
- “If it’s cold **or** raining, I’ll stay home.”

Python does the same—with **logical operators**.

And just like we update things (“add 5 to your score”), Python updates values with **assignment operators**.

This chapter covers:

- Logical Operators: `and`, `or`, `not`
- Assignment Operators: `=`, `+=`, `-=`, etc.
- Truth tables and combined logic
- Practice + mini projects

🔍 Logical Operators

Logical operators let Python **combine multiple conditions**.

Operator	Description	Example	Result
`and`	True if both are True	`True and False`	False
`or`	True if at least one is True	`True or False`	True
`not`	Reverses the result	`not True`	False

Truth Table:

A	B	`A and B`	`A or B`	`not A`
True	True	True	True	False
True	False	False	True	False
False	True	False	True	True
False	False	False	False	True

Code Example:

```
x = 5
y = 10

print(x > 0 and y > 0)    # True
print(x > 0 and y < 0)    # False
print(x > 0 or y < 0)     # True
print(not x > 0)          # False
```

🧠 Nested Logic with Parentheses

Parentheses help you group logic just like math:

```
a = 4
b = 12

print((a > 3 and b < 20) or (a == 2))  # → True
```

🧠 Common Mistake — `=` vs `==`

```
x = 5

# ❌ Wrong
if x = 5:
    print("Yes")   # SyntaxError

# ✅ Correct
if x == 5:
    print("Yes")
```

Use:

- `=` to assign
- `==` to compare

➕ Assignment Operators

Assignment operators are shortcuts for updating variables.

Operator	Action	Example	Equivalent
`=`	Assign	`x = 5`	x becomes 5
`+=`	Add and assign	`x += 3`	`x = x + 3`
`-=`	Subtract	`x -= 2`	`x = x - 2`

Operator	Action	Example	Equivalent
`*=`	Multiply	`x *= 2`	`x = x * 2`
`/=`	Divide	`x /= 3`	`x = x / 3`
`//=`	Floor divide	`x //= 2`	`x = x // 2`
`%=`	Modulus	`x %= 4`	`x = x % 4`
`**=`	Exponent	`x **= 3`	`x = x ** 3`

Code Example:

```
score = 10
score += 5
print(score)  # → 15

score *= 2
print(score)  # → 30
```

🔍 Mini Quiz or Challenge

1. What is `True and False or True`?
2. What does `not (4 > 2)` return?
3. If `x = 3` and `x += 2`, what's the new value of `x`?
4. Fix the bug: `if x = 5:`

💡 Tips and Mistakes

✓ Use `and` when both things must be true
✓ Use `or` when either condition is enough
✓ Use `not` to flip the result
✓ Use assignment shortcuts to write less
✗ Don't confuse `=` and `==`
✗ Don't forget parentheses in complex logic

📜 Summary Recap

- Logical operators: `and`, `or`, `not`
- Used to combine comparisons or conditions
- Assignment operators: `=`, `+=`, `-=`, etc.
- Used to update values quickly

- Use parentheses `()` to group logic
- `=` assigns, `==` compares

🎯 Mini-Project Exercise

Build a pass/fail checker with extra logic

```python
math = int(input("Enter Math marks: "))
science = int(input("Enter Science marks: "))

if math >= 35 and science >= 35:
    print("You passed both subjects!")
else:
    print("You need to work harder.")

# Add bonus
math += 5
print("With bonus, your math marks are:", math)
```

✅ Practice Exercises

🟢 Basic

1. Check if a number is greater than 10 **and** even
2. Use `or` to check if either of two numbers is positive
3. Use `not` to check if a condition is False
4. Start with `x = 10`, then `x += 3`, print final value
5. Show the difference between `=` and `==` in a short snippet

---

🚀 Intermediate

A1. Write a program to check if a number is **between 10 and 20**
A2. Use all three logical operators in one `if` condition
A3. Create a program where the user inputs a number. Add 10 to it using `+=`, then print whether it's now above 50
