6️⃣

Chapter 6

✅ CHAPTER 6 NOTES: Conditionals in Python

🧠 What are Conditionals?

Conditionals allow you to run certain blocks of code **only when a specific condition is true.**

Think of it as making decisions in code — like a "choose your own adventure" book!

🧱 Basic Structure of `if` Statement

```
if condition:
    # Code block runs only if condition is True
else:
    # Code block runs if condition is False
```

📌 Python uses **indentation** (spacing) to decide which code belongs where — no `{}` like in C/C++/Java.

🧱 if-else Ladder

```
if condition1:
    # Executes if condition1 is True
elif condition2:
    # Executes if condition2 is True (only if condition1 is False)
elif condition3:
    # Executes if condition3 is True (if others are False)
else:
    # Executes if NONE of the above conditions are True
```

⛔ **Only one block is executed** in an `if-elif-else` ladder, whichever matches first.

💡 Example 1: Basic Age Checker

```
age = int(input("Enter your age: "))

if age > 18:
    print("You are allowed!")
else:
    print("Not allowed.")
```

💡 Example 2: Complex If-Elif-Else Ladder

```
if age > 18 and age < 110:
    print("You are above 18.")
elif age < 0:
    print("Aliens/time travelers not allowed.")
elif age >= 110:
    print("Ghosts not allowed.")
elif age == 0:
    print("Babies not born yet!")
else:
    print("Invalid entry.")
```

🔎 **Logic Order Matters**

Python checks from top to bottom — first condition that is `True` wins, and remaining are skipped.

✅ `if` vs `elif` vs `else` vs multiple `if`

Type	Behavior
`if`	Checks condition and runs block if True
`elif`	Used after `if` to add more conditions
`else`	Runs only if all previous conditions fail
Another `if`	Starts a new **independent check**, even if earlier if/Elif passed or failed

🧮 Modulo Trick (Even or Odd)

```
if age % 2 == 0:
    print("Even age")
else:
    print("Odd age")
```

This is **very useful** logic in competitive programming for checking parity.

❗ Important Tips & Gotchas

✅ Do This	❌ Avoid This
Use `elif` for multiple options	Don't write multiple `if` if only one should run
Always indent your blocks	Don't forget `:` colon after conditions
Use `and` / `or` for combining conditions	Don't over-nest unless necessary
Use input validation when possible	Don't blindly trust `input()` values

🔀 Logical Operators

Operator	Use	Example
`and`	Both conditions must be True	`if age > 18 and age < 60:`
`or`	At least one must be True	`if age < 0 or age > 150:`

Operator	Use	Example
`not`	Reverses the condition	`if not is_logged_in:`

🧪 Chapter 6 – Practice Problems Explained

✅ Problem 1: Find Greatest of 4 Numbers

```
a, b, c, d = int(input()), int(input()), int(input()), int(input())
e = [a, b, c, d]
e.sort()
print("The largest Number is:", e[3])
```

Concepts Covered:

- List usage for simplification
- `sort()` for finding max
- Alternate (less optimal) approach: using nested `if` statements

📌 Tip: You can also use `max()` for a cleaner solution:

```
print("The largest Number is:", max(a, b, c, d))
```

✅ Problem 2: Pass/Fail Checker

```
# Need at least 33 in each subject and average > 40
if avg > 40 and subject1 > 33 and subject2 > 33 and subject3 > 33:
    print("Passed")
else:
    print("Failed")
```

Concepts Covered:

- Logical `and` operator
- Arithmetic average
- Pass criteria validation

📌 This mimic real-world grading logic, excellent practical implementation.

✅ Problem 3: Spam Detector in Comment

```
# Check for known spam words
if spam1 in comment or spam2 in comment or ...:
    print("Spam")
```

Concepts Covered:

- Use of `in` to check substrings
- Logical `or` operator
- Basic keyword matching (ideal candidate for later AI/NLP extension!)

📌 Better way (for future): Loop over list of spam keywords:

```
spam_words = ["money", "buy now", "subscribe", "click", "earn money"]
if any(word in comment for word in spam_words):
    print("Spam")
```

✅ Problem 4: Username Length Check

```
if len(username) >= 10:
    print("Greater")
else:
    print("Less")
```

Concepts Covered:

- `len()` function
- Simple comparison
- Input validation

✅ Problem 5: Username in List

```
if username in name:
    print("Present")
```

```
else:
    print("Not Present")
```

Concepts Covered:

- `in` keyword for list lookup
- Membership testing

📌 This can later help build user validation or access control systems.

✅ Problem 6: Grade Based on Marks

```
if marks > 90:
    print("Ex")
elif marks > 80:
    print("A")
...
else:
    print("F")
```

Concepts Covered:

- `if-elif-else` ladder
- Real-world grading system
- Percentage to grade conversion

📌 Clean logic and great use of conditional ladders.

✅ Problem 7: Is Post Talking About Harry?

```
if "Harry" in post or "harry" in post:
    print("Yes")
```

Concepts Covered:

- Case sensitivity
- Substring check

📌 Improve it by converting string to lowercase and checking:

```
if "harry" in post.lower():
    print("Yes")
```

🧾 ✅ Chapter 6 – Full Summary Table

Concept	Explanation
`if`, `elif`, `else`	Conditional statements used for decision making
Logical Operators (`and`, `or`)	Used to combine multiple conditions
`in` operator	Checks membership in strings, lists, etc.
`len()` function	Returns length of string or collection
`max()`, `min()`	Can be used to find largest/smallest number
`sort()` and `list`	Used to sort and compare numbers
Case Sensitivity	`"Harry"` is not equal to `"harry"` unless normalized
Substring detection	`if "spam" in comment:` is a powerful way to filter messages
Grade System	Practical use of `if-elif-else` to assign grades
Efficient logic	Prefer using `max()` or lists over multiple `if` comparisons for clean code