🔁 Lecture 03— Loops & Pattern Printing in C++

🧠 **Programming Masterclass ·**

🧠 First Thought Principle — Loop Kya Hota Hai?

Programming me jab bhi ek hi kaam **baar-baar repeat** hota hai,
wahan **Loop** ka concept aata hai.

❓ Real Question

Agar mujhe:

- "Hello" 1 baar print karna ho → easy
- "Hello" 1000 baar print karna ho → ❌ manual impossible

👉 **Yahin loop ka janm hota hai**

🔄 Loop — Basic Definition

**Loop ek aisa structure hai jo kisi block of code ko tab tak repeat karta hai,
jab tak koi condition true rahe.**

👉 Loop = **Controlled Repetition**

🧠 Real-Life Analogy (Very Important)

Socho:

- Alarm tab tak bajta rahe
- Jab tak tum **OFF** na kar do

Yahan:

- Alarm = Loop body
- OFF button = Condition false hona

🔁 Loop Ki Basic Anatomy (DNA of Loop)

Har loop me **sirf 3 cheezein hoti hain**:

1️⃣ **Initialization**
→ Loop kahan se start kare

2️⃣ **Condition**
→ Loop kab tak chale

3️⃣ **Update**
→ Loop ka next step kya hoga

👉 In teenon me se ek bhi galat hua
= ❌ **Infinite loop ya wrong output**

🔂 C++ me Loops Ke Types (Context)

Is lecture me hum mainly **for loop** par focus kar rahe hain
kyunki:

- Patterns
- Counting
- Logic building

👉 Sabse zyada **for loop** se hi bante hain

🔁 for Loop — First Principle View

Syntax:

```
for(initialization; condition; update){

    // body

}
```

🧠 Sochne ka tareeka:

> "Pehle batao start kahan se,
> phir batao kab rukna hai,
> aur har step me kya badlega."

🔍 for Loop Execution — Step by Step

Example:

```
for(int i = 1; i <= 5; i++){
    cout << i << endl;
}
```

Execution Flow:

- Step 1: `i = 1` (initialization)
- Step 2: `i <= 5` → true
- Step 3: body execute → print 1
- Step 4: `i++`
- Step 5: condition check again

👉 Condition false hote hi loop **exit**

🔢 Even Numbers — Two Ways (Logic Building)

✅ Method 1: Direct Jump

```
for(int i = 2; i <= 20; i = i + 2){
    cout << i << endl;
}
```

🧠 Logic:

- Even numbers ka gap = 2
- Fast & clean approach

✅ Method 2: Condition Check

```
for(int i = 1; i <= 20; i++){
    if(i % 2 == 0){
        cout << i << endl;
    }
}
```

🧠 Logic:

- `% 2 == 0` → even
- Flexible approach (filters)

🔢 Odd Numbers Logic

```
for(int i = 1; i <= 20; i = i + 2){
    cout << i << endl;
}
```

👉 Same thinking, bas starting point change

🔤 ASCII Concept — a to z

🧠 First Principle:

`char` internally **numbers (ASCII)** pe kaam karta hai

- `'a'` = 97
- `'b'` = 98

Isliye:

```
for(char ch = 'a'; ch <= 'z'; ch++){
    cout << ch << endl;
}
```

👉 Loop sirf numbers ke liye nahi hota

⚠️ Garbage Value — A Silent Bug

❌ Wrong:

```
int sum;
```

✅ Correct:

```
int sum = 0;
```

🧠 Reason:

- Memory me jo random data hota hai
- wahi variable ke andar aa jata hai

👉 **Initialization is MUST**

➕ Sum of First 10 Natural Numbers

```
int sum = 0;

for(int i = 1; i <= 10; i++){
    sum = sum + i;
}

cout << sum;
```

🧠 Concept:

- Loop = traversal
- `sum` = accumulator

🧠 Mental Model — Yahan Tak Lock Kar Lo

- Loop = controlled repetition
- for loop = most powerful basic tool
- Initialization + Condition + Update = loop backbone
- % operator = filtering logic
- Garbage value = uninitialized variable bug

🧬 Nested Loop — Patterns Ka Foundation

Yahan se programming **sirf syntax nahi**,
Thinking game ban jaati hai.

❓ Core Question

Agar mujhe ek hi kaam:

- sirf repeat nahi
- **rows & columns** ke form me karna ho

to ek loop kaafi hoga?
❌ **Nahi**

👉 Yahin se **Nested Loop** aata hai.

🔁 Nested Loop — Basic Definition

Ek loop ke andar doosra loop chalana
Nested Loop kehlata hai.

👉 Simple language me:

- Loop ke andar loop

🧠 GOLDEN RULE (Lock This)

🔥 **Outer Loop → Rows control karta hai**
🔥 **Inner Loop → Columns control karta hai**

Agar ye rule clear ho gaya,
to **95% patterns automatic solve** ho jaate hain.

📐 Nested Loop Execution — First Thought

Example:

```
for(int row = 1; row <= 3; row++){

    for(int col = 1; col <= 4; col++){

        cout << "* ";

    }

    cout << endl;

}
```

🧠 Execution samjho:

- `row = 1`
    - inner loop 4 baar chalega
- `row = 2`
    - phir inner loop 4 baar
- `row = 3`
    - phir inner loop 4 baar

👉 Inner loop **poora finish hota hai**,
tab outer loop next step leta hai.

⭐ Squar Pattern — First Pattern

Output:

```
*****

*****

*****
```

```
*****

*****
```

Code:

```
for(int row = 0; row < 5; row++){

    for(int col = 0; col < 5; col++){

        cout << "*";

    }

    cout << endl;

}
```

🧠 **Logic:**

- Total rows = 5
- Har row me stars = 5
- Row change → new line

🔢 Number Grid — Row & Column Clarity

Output:

```
1 2 3 4 5

1 2 3 4 5

1 2 3 4 5

1 2 3 4 5

1 2 3 4 5
```

Code:

```
for(int row = 1; row <= 5; row++){

    for(int col = 1; col <= 5; col++){
        cout << col << " ";
    }
    cout << endl;
}
```

🧠 **Key Insight:**

- Column loop decide karta hai **print kya hoga**
- Row loop decide karta hai **kitni baar repeat hoga**

🔤 Character Pattern — ASCII Use

Output:

```
a b c d e
a b c d e
a b c d e
a b c d e
a b c d e
```

Code:

```
for(int row = 1; row <= 5; row++){
    for(char ch = 'a'; ch <= 'e'; ch++){
        cout << ch << " ";
    }
    cout << endl;
}
```

🧠 **Same nested logic**,
sirf datatype change hua.

🧠 Pattern Solving Framework (VIP)

Har pattern ke liye **sirf 3 sawal pucho**:

1️⃣ Total rows kitni hain?
2️⃣ Har row me kya print ho raha hai?
3️⃣ Row change kab ho rahi hai?

👉 Answer mil gaya = pattern solved.

🔁 Simple Loop Revision

```
for(int i = 1; i <= 5; i++){

    cout << "Hello Coder Army" << endl;

}
```

🧠 Loop sirf numbers ke liye nahi,
kisi bhi repeatable task ke liye hota hai.

🔄 Reverse Loop — Thinking Expander

```
for(int i = 10; i >= 1; i--){

    cout << i << " ";

}
```

🧠 Loop ka direction bhi control me hota hai.

🧠 Mental Model — Yahan Tak Lock Kar Lo

- Nested loop = loop inside loop
- Outer loop → rows
- Inner loop → columns
- Pattern = rows × columns
- Datatype change se pattern nature change hota hai

⭐ Triangle Patterns — Real Logic Starts Here

Ab tak hum **rows × columns** wale fixed patterns kar rahe the.
Triangle patterns me **columns har row ke saath change hote hain**.

👉 **Yahin se thinking level upgrade hota hai.**

🧠 First Thought — Triangle Kya Hota Hai?

Triangle ka matlab:

- Row 1 → kam output
- Row 2 → thoda zyada
- Har next row → growth

👉 Pattern = **row-dependent printing**

⭐ Right Triangle — First Growth Pattern

Output:

```
*

**

***

****

*****
```

🧠 Soch ka tareeka

- Total rows = 5
- Row 1 → 1 star
- Row 2 → 2 stars
- Row N → N stars

👉 **Stars = Row number**

Code:

```
for(int row = 1; row <= 5; row++){

    for(int col = 0; col < row; col++){

        cout << "*";

    }

    cout << endl;

}
```

⭐ Inverted Triangle — Decreasing Logic

Output:

```
*****

****

***

**

*
```

🧠 Soch ka tareeka

- Total rows = 5
- Row 1 → 5 stars
- Row 2 → 4 stars
- Row N → (totalRows − row + 1) stars

Code:

```
for(int row = 5; row >= 1; row--){
    for(int col = 0; col < row; col++){
        cout << "*";
    }
    cout << endl;
}
```

🧠 Growing vs Shrinking Logic

- Right Triangle → **inner loop depends on row (increase)**
- Inverted Triangle → **inner loop depends on row (decrease)**

👉 Bas **direction change**, logic same.

⭐ Right Aligned Triangle — Space + Star Game

Output:

```
    *
   **
  ***
 ****
*****
```

🧠 First Principle Soch

Yahan do cheezein hoti hain:
1️⃣ **Spaces**
2️⃣ **Stars**

👉 Dono ka relation samajhna hi pattern ka master-key hai.

🧠 Row-wise Breakdown (5 rows)

Row	Spaces	Stars
1	4	1
2	3	2
3	2	3
4	1	4
5	0	5

Code:

```
for(int row = 4; row >= 0; row--){

    for(int space = 0; space < row; space++){

        cout << " ";

    }

    for(int star = 0; star < 5 - row; star++){

        cout << "*";

    }

    cout << endl;

}
```

🧠 Pattern Logic — Universal Framework

Har pattern solve karne se pehle:
1️⃣ Rows count karo
2️⃣ Har row ke liye:

- Spaces kitni?
- Symbols kitne?
  3️⃣ Relation likh lo (formula)

👉 Code baad me aata hai,
logic pehle aata hai.

🔁 Table Program — Loop in Real Use

```
int table = 7;


for(int i = 1; i <= 10; i++){

    cout << table * i << endl;

}
```

🧠 Loop ka practical use — **multiplication tables**

🧠 Yahan Tak Lock Kar Lo

- Triangle = row-dependent logic
- Inner loop ka limit = pattern ka soul
- Spaces bhi output ka part hain
- Pehle table banao, phir code likho

🔢 Continuous Number Grid — The Most Intelligent Pattern

Yahan par patterns **sirf printing nahi** rehte,
yahan **state management** start hoti hai.

❓ Core Question

Agar mujhe ye print karna ho:

```
1    2    3    4    5

6    7    8    9    10

11   12   13   14   15

16   17   18   19   20

21   22   23   24   25
```

to simple `row` aur `col` se kaam ho jayega?
❌ **Nahi**

👉 Kyunki numbers **reset nahi hone chahiye**.

🧠 First Thought — Counter Kya Hota Hai?

Counter ek aisa variable hota hai:

- jo **value yaad rakhta hai**
- aur har print ke baad **update hota hai**

👉 Pattern ka brain = **counter placement**

⚠️ Common Mistake (VERY IMPORTANT)

❌ Galat soch:

```
for(row){

    int num = 1;    // ❌ yahin galti

    for(col){

        cout << num++;

    }

}
```

Result:

- Har row me number **1 se reset** ho jaayega

✅ Correct Strategy — Counter Outside

```
int num = 1;


for(int row = 0; row < 5; row++){

    for(int col = 0; col < 5; col++){

        cout << num << " ";

        num++;

    }

    cout << endl;

}
```

🧠 Why This Works (First Principle)

- `num` outer loop ke bahar hai
- Isliye value **persist** karti hai
- Har inner iteration me increment hota hai
- Kabhi reset nahi hota

👉 **State preserved**

🧠 Pattern Decision Tree (GOLD)

Har pattern se pehle khud se poochho:

1️⃣ Kya printing repeat ho rahi hai? → Loop
2️⃣ Kya rows & columns hain? → Nested loop
3️⃣ Kya value reset nahi honi chahiye? → Counter outside

👉 Ye 3 sawal = 90% pattern solved

🔁 Pattern vs Program Thinking

- Pattern = visual logic
- Program = state + condition + update

👉 Continuous number grid
pattern + program thinking ka bridge hai

🧠 Mental Model — This Is Critical

- Counter ko bahar rakhna = memory control
- Inner loop = kitna print hoga
- Outer loop = kitni rows
- Reset ka control = bug vs correct output

🔒 Yahan Tak Lock Kar Lo

- Continuous patterns me **counter bahar**
- Variable scope matters
- Pattern sirf design nahi, **state management** hai

🧠 Pattern Thinking — The Real Game

Agar tum sirf code yaad kar rahe ho,
to tum **pattern solve nahi kar rahe**,
tum **copy kar rahe ho.**

Pattern solving ka asli matlab hai:
👉 **Sochna before coding**

🔑 Universal Pattern Framework (GOLD)

Har pattern se pehle ye 5 sawal apne aap se poochho:

1️⃣ Total **rows** kitni hain?
2️⃣ Har row me **kitna print** ho raha hai?
3️⃣ Kya output **row-dependent** hai?
4️⃣ Kya koi **counter/state** chahiye?
5️⃣ Kya koi value **reset nahi honi** chahiye?

👉 Inka answer mil gaya = **pattern solved**

🔁 Loop Mental Model (Lock This)

🔷 for Loop ka DNA

```
Start → Condition → Body → Update → Repeat
```

Agar kisi bhi jagah:

- start galat
- condition galat
- update missing

👉 ❌ Infinite loop / wrong output

🧬 Nested Loop Rule (95% Patterns)

- **Outer Loop → Rows**
- **Inner Loop → Columns**

Ye rule yaad nahi,
feel hona chahiye.

⭐ Star Patterns — Master Key

Right Triangle

- Stars = row number

Inverted Triangle

- Stars = totalRows − row + 1

Right Aligned Triangle

- Spaces + Stars
- Spaces decrease, stars increase

👉 Har pattern = **math relation**

🔢 Number Patterns — State Management

Static Number Grid

- Inner loop decides **what to print**

Continuous Number Grid

- Counter **outer loop ke bahar**
- Reset = bug

⚠️ Common Mistakes (Exam + Interview)

❌ Counter ko galat scope me rakhna
❌ Spaces ko ignore karna
❌ Row vs column ka confusion
❌ Variable initialize na karna (garbage value)
❌ Logic soche bina code likhna

🧠 Interview Battle Checklist

Exam me pattern aaya:
1️⃣ Rough me rows likho
2️⃣ Har row ka output likho
3️⃣ Relation derive karo
4️⃣ Loops likho
5️⃣ Code clean rakho

👉 Panic = fail
👉 Process = pass

🧠 Interview Truth

Interviewer code se zyada dekhta hai:

- Tum kaise **approach** karte ho
- Tum kaise **explain** karte ho
- Tum logic ko kaise **break** karte ho

Agar tum bol pa rahe ho:

> "Outer loop rows control karta hai,
> inner loop columns,
> aur counter bahar isliye rakha kyunki value reset nahi chahiye"

👉 **Game over. Selected.**

🔒 Final Lock — Yah Lecture Kya Sikhata Hai

- Loop = repetition with control
- Pattern = logic visualization
- Code = logic ka translation
- Practice = clarity

✨ Made By @Harshal Chauhan ✨