Day 3: Loops & Iteration

لوپس اور دہرانا

> **Quote of the Day:** "*Repetition is the mother of learning, the father of action, which makes it the architect of accomplishment.*" - Zig Ziglar
>
> *"دہرانا سیکھنے کی ماں، عمل کا باپ، اور کامیابی کا معمار ہے۔"*

📋 Today's Learning Goals (آج کے اہداف)

By the end of today, you will:

- ☐ Master for loops and understand their 3 parts
- ☐ Use while loops for conditional repetition
- ☐ Know when to use break and continue
- ☐ Avoid infinite loops (the programmer's nightmare!)
- ☐ Build a Cricket Score Counter that simulates a match

Time Breakdown (کل وقت: 150 منٹ)

- 🕐 7:00-7:05 PM (5min): Standup - Share your Biryani Checker results!
- 🕐 7:05-8:05 PM (60min): Understanding loops (3× Pomodoro)
- 🕐 8:05-8:50 PM (45min): Practice with different loop patterns
- 🕐 8:50-9:25 PM (35min): Cricket Score Counter project
- 🕐 9:25-9:30 PM (5min): Quiz & reflection

🎯 What We're Building Today

Today you'll create a **Cricket Score Counter** - a program that simulates a cricket innings, generates random runs for each ball, tracks boundaries, calculates strike rate, and can even handle wickets!

Why This Matters for Your Career: Every app uses loops constantly:

- Instagram: Loops through posts to display your feed
- Daraz: Loops through products to show search results

- WhatsApp: Loops through messages to show chat history
- Google Maps: Loops through routes to find the best one

Today you're learning how to make computers repeat tasks efficiently!

🧠 سمجھنا (Understanding): Why Loops Exist

The Real-World Analogy

Scenario: Reciting Tasbih after Namaz

Imagine you need to say "SubhanAllah" 33 times:

Without loops (manual way):

```
Say "SubhanAllah"  // 1
Say "SubhanAllah"  // 2
Say "SubhanAllah"  // 3
... (30 more times!)
Say "SubhanAllah"  // 33
```

With loops (smart way):

```
Count from 1 to 33:
    Say "SubhanAllah"
```

This is exactly what loops do in programming!

Daily Life Examples

You use loops every day without realizing:

1. **Making Roti:**
    - For each person at dinner (loop through family)
    - Make one roti
    - Repeat until everyone has enough
2. **Cricket Overs:**
    - For each over (1 to 20)
    - For each ball (1 to 6)
    - Bowl the ball

    - Repeat

3. **Checking Exam Papers:**

    - For each student's paper (loop through stack)
    - Calculate marks
    - Assign grade
    - Next paper

Why Does This Matter?

Without loops, you'd need to write the same code hundreds or thousands of times!

Example: Printing numbers 1 to 100

```
// Without loop (□□□□□□□!)
console.log(1);
console.log(2);
console.log(3);
// ... 97 more lines!
console.log(100);

// With loop (□□□□□!)
for (let i = 1; i <= 100; i++) {
    console.log(i);
}
// Just 3 lines! 🎉
```

The Mental Model

Think of a loop like a **roundabout (چکر)** in traffic:

```
Enter roundabout → Take one lap → Check: Done?
                                    ↓ No
                              ← Continue ←
                                    ↓ Yes
                                   Exit
```

Your code goes in circles until the condition says "STOP!"

📚 Building Block #1: for Loop (بنیادی لوپ)

What is a for Loop? (کیا ہے؟)

Urdu Analogy: Think of a for loop like counting **tasbeeh beads on a tasbih** (تسبیح کے دانے گننا).

You know:

- **Start:** Begin at bead 1
- **End:** Stop at bead 33
- **Action:** Say "SubhanAllah" for each bead

In JavaScript:

```
for (let bead = 1; bead <= 33; bead++) {
    console.log("SubhanAllah");
}
```

The Three Parts of a for Loop

```
for (initialization; condition; increment) {
    // Code to repeat
}
```

Part	What It Does	Example	Urdu
Initialization	Start point	`let i = 1`	شروعات
Condition	When to stop	`i <= 10`	شرط
Increment	How to move forward	`i++`	اضافہ

How It Works - Step by Step

```
// THINKING: Count from 1 to 5

for (let i = 1; i <= 5; i++) {
    console.log(i);
}

// What happens:
// Step 1: let i = 1          (Create counter, set to 1)
// Step 2: Check i <= 5?      (Is 1 <= 5? Yes!)
// Step 3: console.log(1)     (Print 1)
// Step 4: i++                (i becomes 2)
// Step 5: Check i <= 5?      (Is 2 <= 5? Yes!)
// Step 6: console.log(2)     (Print 2)
// ... continues until i = 6
// Step N: Check i <= 5?      (Is 6 <= 5? No! STOP)
```

Output:

```
1
2
3
4
5
```

The i++ Operator

```
// i++ means "add 1 to i"
let i = 5;
i++;         // i is now 6
i++;         // i is now 7

// Same as:
i = i + 1;

// Other variations:
i += 2;      // Add 2 to i
i--;         // Subtract 1 from i (□□□□□□)
```

Your First Example

```
// THINKING: Print even numbers from 2 to 10

for (let num = 2; num <= 10; num += 2) {
    console.log(num);
}
// Output: 2, 4, 6, 8, 10

// TODO: Print odd numbers from 1 to 9
for (let num = _____; num <= _____; num += _____) {
    console.log(num);
}

// TODO: Count backwards from 10 to 1
for (let num = _____; num >= _____; num_____) {
    console.log(num);
}
```

Common Patterns

1. Standard Forward Loop:

```
// Count 1 to 10
for (let i = 1; i <= 10; i++) {
    console.log(i);
}
```

2. Backward Loop:

```
// Countdown 10 to 1
for (let i = 10; i >= 1; i--) {
    console.log(i);
}
```

3. Skip Pattern:

```
// Every 5th number: 0, 5, 10, 15, 20
for (let i = 0; i <= 20; i += 5) {
    console.log(i);
}
```

4. Loop Through Range:

```
// Multiply by 5: table of 5
for (let i = 1; i <= 10; i++) {
    console.log(`5 × ${i} = ${5 * i}`);
}
```

Common Mistakes

❌ Wrong:

```
for (let i = 1; i <= 10) {  // Missing increment!
    console.log(i);
}
// INFINITE LOOP! i never changes!
```

✅ Right:

```
for (let i = 1; i <= 10; i++) {  // Has increment
    console.log(i);
}
```

❌ Wrong:

```
for (let i = 1; i <= 10; i++) {
    console.log(j);  // Wrong variable! Should be i
}
```

✅ **Right:**

```
for (let i = 1; i <= 10; i++) {
    console.log(i);  // Correct variable
}
```

❌ **Wrong:**

```
for (let i = 10; i <= 1; i++) {  // Will never run!
    console.log(i);  // 10 is not <= 1
}
```

✅ **Right:**

```
for (let i = 10; i >= 1; i--) {  // Correct condition
    console.log(i);
}
```

Check Your Understanding

- [ ] What are the 3 parts of a for loop?
- [ ] What does `i++` mean?
- [ ] How do you loop backwards?
- [ ] What makes a loop run forever?

Quick Test:

```
// What will this print?
for (let i = 0; i < 3; i++) {
    console.log(i);
}
// Your answer: _____________

// What about this?
for (let i = 5; i > 2; i--) {
    console.log(i);
}
// Your answer: _____________
```

🧱 Building Block #2: while Loop (جب تک لوپ)

What is a while Loop? (کیا ہے؟)

Urdu Analogy: Think of waiting in line at **National Bank** for bill payment.

```
WHILE (قطار میں لوگ ہیں) {
    انتظار کرو
    آگے بڑھو
}
```

You don't know HOW MANY people are ahead. You just keep waiting UNTIL it's your turn!

How It Works - Step by Step

```
// THINKING: Keep asking until correct password

let password = "";

while (password !== "12345") {
    password = prompt("Enter password:");
    // Note: prompt() only works in browser
}

console.log("Access granted! ✅");
```

The pattern:

```
while (condition is true) {
    // Keep doing this
    // Eventually condition becomes false
}
```

for Loop vs while Loop

Use for when: You know HOW MANY times to repeat

```
// Print 1 to 10 (I know: 10 times!)
for (let i = 1; i <= 10; i++) {
    console.log(i);
}
```

Use while when: You don't know how many times, just the condition

```
// Keep rolling dice until you get 6
let roll = 0;
while (roll !== 6) {
    roll = Math.floor(Math.random() * 6) + 1;
    console.log("Rolled:", roll);
}
console.log("Got 6! 🎉");
```

Your First Example

```
// THINKING: Find first number divisible by 7 after 50

let num = 51;

while (num % 7 !== 0) {
    num++;
}

console.log("First number divisible by 7 after 50:", num);
// Output: 56

// TODO: Find first number > 100 divisible by 11
let number = 101;

while (number _____ 11 _____ 0) {
    number++;
}

console.log("Answer:", number);
```

Real-World Example

```
// THINKING: Careem driver searching for passenger

let driverFound = false;
let searchTime = 0;

while (!driverFound && searchTime < 5) {
    console.log("Searching for driver... منتظر رہیں");
    searchTime++;

    // Simulate: 60% chance of finding driver
    if (Math.random() > 0.4) {
        driverFound = true;
    }
}

if (driverFound) {
    console.log("Driver found! ڈرائیور مل گیا ✅");
} else {
    console.log("No driver available. دوبارہ کوشش کریں");
}
```

Common Mistakes

❌ Wrong:

```
let i = 1;
while (i <= 5) {
    console.log(i);
    // Forgot to increment!
}
// INFINITE LOOP! i stays 1 forever
```

✅ Right:

```
let i = 1;
while (i <= 5) {
    console.log(i);
    i++;  // Must update the variable!
}
```

Check Your Understanding

- ☐ When should you use while instead of for?
- ☐ What makes a while loop stop?
- ☐ What's the danger of while loops?
- ☐ Can a while loop run zero times?

📚 Building Block #3: break and continue

What are break and continue? (کیا ہیں؟)

Cricket Analogy:

break = **All out!** (سب آؤٹ!)

```
Team is batting
Wicket falls
If 10 wickets down → STOP innings (break!)
```

continue = **No run** (کوئی رن نہیں)

```
Ball bowled
If wide/no-ball → Skip, bowl next ball (continue!)
```

break - Exit the Loop Immediately

```
// THINKING: Find first number > 50 divisible by 7

for (let i = 1; i <= 100; i++) {
    if (i > 50 && i % 7 === 0) {
        console.log("Found it:", i);
        break;  // Stop searching! We found it
    }
}
// Output: Found it: 56
// Loop stops immediately, doesn't continue to 100
```

continue - Skip to Next Iteration

```
// THINKING: Print odd numbers 1-10 (skip even)

for (let i = 1; i <= 10; i++) {
    if (i % 2 === 0) {
        continue;  // Skip even numbers
    }
    console.log(i);  // Only odd numbers reach here
}
// Output: 1, 3, 5, 7, 9
```

Visual Difference

```
// WITH break - stops completely
for (let i = 1; i <= 5; i++) {
    if (i === 3) break;
    console.log(i);
}
// Output: 1, 2  (stops at 3)

// WITH continue - skips one iteration
for (let i = 1; i <= 5; i++) {
    if (i === 3) continue;
    console.log(i);
}
// Output: 1, 2, 4, 5  (skips 3, continues to 5)
```

Real Example: Search with Limit

```
// THINKING: Search for student by roll number (max 100 tries)
```

```
const students = ["Ali", "Sara", "Ahmed", "Fatima", "Hassan"];
const searchFor = "Ahmed";
let found = false;

for (let i = 0; i < students.length; i++) {
    if (students[i] === searchFor) {
        console.log("Found at position:", i);
        found = true;
        break;  // Stop searching once found!
    }
}

if (!found) {
    console.log("Student not found");
}
```

Your First Example

```
// TODO: Print numbers 1-20 but skip multiples of 3

for (let i = 1; i <= 20; i++) {
    // HINT: Use continue when i is divisible by 3
    if (i _____ 3 === 0) {
        continue;
    }
    console.log(i);
}

// TODO: Find first number between 100-200 divisible by 13
for (let num = 100; num <= 200; num++) {
    if (num _____ 13 === 0) {
        console.log("Answer:", num);
        _____;  // Stop once found
    }
}
```

Check Your Understanding

- ☐ What does break do?
- ☐ What does continue do?
- ☐ When would you use break?
- ☐ When would you use continue?

Building Block #4: Avoiding Infinite Loops

What is an Infinite Loop? (لامحدود لوپ)

Urdu Analogy: Like being stuck in **I.I. Chundrigar Road traffic** that never ends! 🚗

An infinite loop runs FOREVER because the condition NEVER becomes false.

Common Causes

1. Forgot to Update Counter:

```
// ❌ INFINITE LOOP!
let i = 1;
while (i <= 5) {
    console.log(i);
    // Forgot i++
}
// i stays 1, condition always true!
```

2. Wrong Condition:

```
// ❌ INFINITE LOOP!
for (let i = 1; i >= 0; i++) {
    console.log(i);
}
// i keeps increasing, always >= 0!
```

3. Update Goes Wrong Direction:

```
// ❌ INFINITE LOOP!
for (let i = 10; i > 0; i++) {  // Going up!
    console.log(i);
}
// i increases, never reaches 0!
```

How to Prevent

✅ **Safety Check #1:** Always update your counter

```
let i = 1;
while (i <= 5) {
    console.log(i);
    i++;  // MUST HAVE THIS!
}
```

✅ **Safety Check #2:** Condition must eventually be false

```
for (let i = 1; i <= 10; i++) {
    // i will reach 11, then i <= 10 is false ✅
}
```

✅ **Safety Check #3:** Add safety limit

```
let tries = 0;
while (condition && tries < 100) {  // Max 100 iterations
    // Your code
    tries++;
}
```

Emergency: How to Stop Infinite Loop

If your browser freezes:

1. **Close the tab** (Ctrl + W)
2. **Open Task Manager** (Ctrl + Shift + Esc)
3. **End browser process**
4. **Fix your code** before running again!

Check Your Understanding

- [ ] What is an infinite loop?
- [ ] What causes infinite loops?
- [ ] How do you prevent them?
- [ ] What do you do if code freezes?

💻 Practice Session: Loop Mastery

🎯 Practice Goal

By the end of this section, you'll confidently write any type of loop!

Exercise 1: Multiplication Table (ہم ساتھ کریں)

Scenario: Create a multiplication table for any number

Starter Code:

```
// TODO Step 1: Choose a number
const number = 7;

console.log(`Table of ${number}:`);
console.log("=================");

// TODO Step 2: Loop 1 to 10
for (let i = _____; i <= _____; i++) {
    // TODO Step 3: Calculate and display
    const result = number _____ i;
    console.log(`${number} × ${i} = ${result}`);
}

// Expected output:
// 7 × 1 = 7
// 7 × 2 = 14
// ... etc
```

Test Your Code: Try with different numbers: 2, 5, 12, 19

Exercise 2: Sum Calculator (آپ اب)

Problem: Calculate sum of numbers from 1 to N

Requirements:

- [ ] Ask for a number N
- [ ] Add all numbers from 1 to N
- [ ] Display total sum

Thinking Framework:

1. What variable do I need? (sum, starting at 0)
2. How do I loop from 1 to N?
3. What happens each iteration? (add current number to sum)

Starter Code:

```
const N = 10;  // Try different values

let sum = 0;  // Start with zero

// TODO: Loop from 1 to N
for (let i = _____; i <= _____; i++) {
    // TODO: Add i to sum
    sum _____ i;
}

console.log(`Sum of 1 to ${N} is: ${sum}`);

// Test:
// N = 5  → Answer should be 15 (1+2+3+4+5)
// N = 10 → Answer should be 55
```

Don't Look Below Until You Try! ⬇️

Hints (if stuck):

▶ Stuck on the loop?

```
for (let i = 1; i <= N; i++) {
    sum += i;  // Same as sum = sum + i
}
```

Exercise 3: Even Number Counter

Problem: Count how many even numbers between 1 and 50

```
let count = 0;

// TODO: Loop through 1 to 50
for (let i = _____; i <= _____; i++) {
    // TODO: Check if even
    if (i _____ 2 === 0) {
        count++;  // Increment counter
    }
}

console.log("Even numbers from 1-50:", count);
// Answer should be: 25
```

Exercise 4: Find Multiples

Problem: Find all multiples of 7 between 1 and 100

```
console.log("Multiples of 7:");

for (let num = 1; num <= 100; num++) {
    if (num _____ 7 === 0) {
        console.log(num);
    }
}

// Output: 7, 14, 21, 28, ... 98
```

Exercise 5: Countdown Timer

Problem: Create a countdown from 10 to 1

```
console.log("🚀 Launch Countdown:");

for (let i = _____; i _____ 1; i_____) {
    console.log(i + "...");
}

console.log("💥 Blast off!");

// Output:
// 10...
// 9...
// 8...
// ...
// 1...
// 💥 Blast off!
```

Exercise 6: Password Attempts (while loop)

Problem: Give user 3 chances to enter correct password

```
const correctPassword = "Pakistan123";
let attempts = 0;
let maxAttempts = 3;
let userPassword = "";

while (userPassword !== correctPassword && attempts < maxAttempts) {
    // In real code, use prompt()
    // For testing, you can manually change userPassword

    userPassword = "wrongpassword";  // Change this to test
    attempts++;

    console.log(`Attempt ${attempts} of ${maxAttempts}`);
}

if (userPassword === correctPassword) {
    console.log("✅ Login successful!");
} else {
    console.log("❌ Account locked. Too many attempts.");
}
```

🚀 آج کا چیلنج (Today's Challenge)

Project: Cricket Score Counter

کرکٹ سکور کاؤنٹر

The Problem: You're scoring a T20 cricket match! Simulate a complete innings where:

- Team bats for 6 overs (each over = 6 balls)
- Each ball generates random runs (0-6)
- Track boundaries (4s and 6s)
- Calculate total runs and strike rate
- Can get out randomly

What You're Building: A program that simulates live cricket scoring with ball-by-ball updates!

Success Criteria:

- ☐ Loops through 6 overs correctly
- ☐ Each over has exactly 6 balls
- ☐ Generates random runs (0-6)
- ☐ Tracks boundaries correctly
- ☐ Calculates total and strike rate
- ☐ Shows commentary
- ☐ No console errors

Phase 1: Planning (پہلے سوچیں)

Before coding, answer:

1. **How many balls total?**
    - 6 overs × 6 balls = 36 balls
2. **What data do I need to track?**
    - Total runs
    - Number of 4s

    - Number of 6s
    - Balls faced
    - Wickets fallen

3. **How do I generate random runs?**
    - Use Math.random() to get 0-6
4. **What's strike rate?**
    - Strike rate = (Total runs / Balls faced) × 100

Planning Checkpoint:

- ☐ I understand the loop structure (outer: overs, inner: balls)
- ☐ I know how to generate random numbers
- ☐ I know what variables to track
- ☐ I understand the calculations

Phase 2: Foundation (بنیاد)

Starter Code:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Cricket Score Counter</title>
</head>
<body>
    <h1>🏏 Cricket Score Counter</h1>
    <h2>Pakistan Innings Simulation</h2>
    <h3>Press F12 to see ball-by-ball commentary!</h3>

    <script>
        // ===================================
        // CRICKET SCORE COUNTER
        // By: [Your Name]
        // Date: [Today's Date]
        // ===================================

        console.log("🏏 کرکٹ سکور کاؤنٹر");
        console.log("============================");
        console.log("Pakistan vs India - T20 Match");
        console.log("Pakistan Innings");
        console.log("============================\n");

        // TODO Step 1: Initialize tracking variables
        let totalRuns = 0;
        let boundaries4 = 0;  // Count of 4s
        let boundaries6 = 0;  // Count of 6s
        let ballsFaced = 0;
        let wickets = 0;
        const maxWickets = 10;

        // TODO Step 2: Loop through overs
        // HINT: Outer loop for overs (1 to 6)
        for (let over = 1; over _____ 6; over++) {

            console.log(`\n--- Over ${over} ---`);

            // TODO Step 3: Loop through balls in each over
            // HINT: Inner loop for balls (1 to 6)
            for (let ball = 1; ball _____ 6; ball++) {

                // Check if all out
                if (wickets _____ maxWickets) {
                    console.log("\n❌ ALL OUT! سب آؤٹ!");
                    _____; // Exit loop
                }

                // TODO Step 4: Generate random runs (0-6)
                // HINT: Math.random() gives 0 to 0.999...
                // Multiply by 7 to get 0 to 6.999...
                // Math.floor() to round down to integer
                const runs = Math.floor(Math.random() _____ 7);

                // TODO Step 5: Add to total
                totalRuns _____ runs;
                ballsFaced++;

                // TODO Step 6: Check for boundaries
                if (runs === 4) {
                    boundaries4++;
                    console.log(`Ball ${ball}: FOUR! چوکا! 🎉`);
                } else if (runs _____ 6) {
                    boundaries6++;
                    console.log(`Ball ${ball}: SIX! چھکا! 🚀 What a shot!`);
                } else if (runs === 0) {
                    console.log(`Ball ${ball}: Dot ball - No run`);
                } else {
                    console.log(`Ball ${ball}: ${runs} run(s)`);
                }

                // TODO BONUS: Random wicket (10% chance)
```

```
                // HINT: if (Math.random() < 0.1) means 10% chance
                if (Math.random() < 0.1 && wickets < maxWickets) {
                    wickets++;
                    console.log(`💥 WICKET! آؤٹ! Wickets: ${wickets}/${maxWickets}`);
                }
            }

            // End of over summary
            console.log(`End of Over ${over}: ${totalRuns}/${wickets}`);
        }

        // TODO Step 7: Calculate strike rate
        // Strike rate = (runs / balls) × 100
        const strikeRate = (totalRuns / ballsFaced) _____ 100;

        // TODO Step 8: Display final scorecard
        console.log("\n============================");
        console.log("📊 FINAL SCORECARD");
        console.log("============================");
        console.log(`Total Runs: ${totalRuns}`);
        console.log(`Wickets: ${wickets}`);
        console.log(`Balls Faced: ${ballsFaced}`);
        console.log(`Fours: ${boundaries4} × 4 = ${boundaries4 * 4} runs`);
        console.log(`Sixes: ${boundaries6} × 6 = ${boundaries6 * 6} runs`);
        console.log(`Strike Rate: ${strikeRate.toFixed(2)}`);
        console.log("============================");

        // TODO Step 9: Determine result
        if (wickets === maxWickets) {
            console.log("❌ Team All Out!");
        } else {
            console.log("✅ Innings Complete!");
        }

        // TODO BONUS: Add target message
        console.log(`\nTarget for India: ${totalRuns + 1} runs to win`);

    </script>
</body>
</html>
```

Phase 3: Milestones (سنگ میل)

Milestone 1: Basic Loop Works ✅

- ☐ Outer loop runs 6 times (overs)
- ☐ Inner loop runs 6 times per over

- [ ] Can see ball count in console
- Test: Count total console.logs - should be 36 balls

Milestone 2: Random Runs Generated ✅

- [ ] Each ball shows a number 0-6
- [ ] Runs add to total correctly
- [ ] No errors in calculations
- Test: Total runs should be between 0-216 (6×36)

Milestone 3: Boundaries Counted ✅

- [ ] Detects 4s correctly
- [ ] Detects 6s correctly
- [ ] Special messages for boundaries
- Test: Run multiple times, 4s and 6s should vary

Milestone 4: Complete Scorecard ✅

- [ ] Strike rate calculates correctly
- [ ] All statistics display
- [ ] Professional looking output
- Test: Does output look like real cricket score?

Debugging Guide (اگر پھنس جائیں)

Problem: Infinite loop (page freezes)

- [ ] Check: Do both loops have increment (over++, ball++)?
- [ ] Check: Are conditions correct (<= not >=)?
- [ ] Close tab immediately (Ctrl + W)

Problem: Wrong number of balls

- [ ] Check: Inner loop should be `ball <= 6` not `ball < 6`
- [ ] Check: Are loops nested correctly?
- Add: console.log to count total balls

Problem: Random runs not working

- [ ] Check: `Math.floor(Math.random() * 7)` not `Math.random() * 7`
- [ ] Check: Math.random() gives 0-0.999, need to multiply by 7
- Test: console.log the random number each time

Problem: Strike rate shows NaN

- ☐ Check: Did you increment ballsFaced each ball?
- ☐ Check: Are you dividing by ballsFaced not 0?
- ☐ Check: Did you use * 100 not × 100?

Common Logic Issues:

```
// ❌ WRONG: Boundary counts wrong
if (runs = 4) {  // Assignment, not comparison!
    boundaries4++;
}

// ✅ RIGHT:
if (runs === 4) {  // Comparison
    boundaries4++;
}
```

Extension Challenges (بونس چیلنج)

If you finish early:

✨ Level 1: Add Over Summaries

```
// At end of each over, show runs scored in that over
let overRuns = 0;  // Reset each over
// Track runs per over
// Display at end of each over
```

✨✨ Level 2: Calculate Run Rate

```
// Run rate = Total runs / Number of overs
// Show current run rate
// Compare to required rate if chasing
```

✨✨✨ Level 3: Player Statistics

```
// Track two batsmen
// Show individual scores
// Show partnerships
// Handle wickets properly
const batsman1 = { name: "Babar", runs: 0, balls: 0 };
const batsman2 = { name: "Rizwan", runs: 0, balls: 0 };
```

📝 Daily Quiz (5 منٹ کا ٹیسٹ)

Instructions: Answer WITHOUT looking at notes!

1. What are the three parts of a for loop?

- A) start, middle, end
- B) initialization, condition, increment
- C) begin, check, update
- D) setup, test, loop

▶ See Answer (Try first!)

Answer: B - initialization (شروعات), condition (شرط), increment (اضافہ). Example: `for (let i = 1; i <= 10; i++)` where `i = 1` is initialization, `i <= 10` is condition, `i++` is increment.

---

2. What does `i++` do?

- A) Multiplies i by 2
- B) Adds 1 to i
- C) Subtracts 1 from i
- D) Does nothing

▶ See Answer (Try first!)

Answer: B - Adds 1 to i. It's shorthand for `i = i + 1`. If i is 5, after `i++`, i becomes 6.

---

3. What will this code output?

```
for (let i = 3; i > 0; i--) {
    console.log(i);
}
```

- A) 3, 2, 1
- B) 0, 1, 2, 3
- C) 1, 2, 3
- D) Nothing

▶ See Answer (Try first!)

Answer: A - 3, 2, 1. The loop starts at 3, runs while i > 0, and decreases each time (i--). So it prints 3, then 2, then 1, then stops (0 is not > 0).

---

4. What does `break` do in a loop?

- A) Pauses the loop temporarily
- B) Skips to the next iteration
- C) Exits the loop immediately
- D) Restarts the loop

▶ See Answer (Try first!)

Answer: C - Exits the loop immediately. Think "ALL OUT!" in cricket - the innings stops completely, doesn't continue to the next ball.

---

5. Which creates an infinite loop?

- A) `for (let i = 1; i <= 10; i++) {}`
- B) `while (true) {}`
- C) `for (let i = 0; i < 5; i++) {}`
- D) `while (false) {}`

▶ See Answer (Try first!)

Answer: B - `while (true) {}` because the condition is ALWAYS true, it never stops! Like traffic that never ends. Always make sure your loop condition can eventually become false!

Scoring:

- **5/5:** 🎉 Loop Master! You're ready for complex iterations!
- **4/5:** 👏 Great! Review the one you missed
- **3/5:** 👍 Good progress! Practice more loops
- ❤️ **/5:** 💪 Review all loop concepts again

🎓 Today's Homework (گھر کا کام)

Required (لازمی):

- [ ] Complete the Cricket Score Counter
- [ ] Run it 5 times to see different results
- [ ] Explain to a family member how loops work using tasbih beads

Optional (اختیاری):

- ☐ Try the extension challenges
- ☐ Create a "Countdown Timer" (10 to 1)
- ☐ Make a "Times Table Generator" (ask for number, show table)
- ☐ Build a "Sum Calculator" (sum from 1 to N)

For Tomorrow:

- ☐ Think about: "How would I create reusable code that I can call multiple times?"
- ☐ This will help with tomorrow's topic: Functions!

💭 Daily Reflection (روزانہ کی سوچ)

آج میں نے کیا سیکھا (What I Learned Today):

مشکل کیا لگا (What I Found Difficult):

مزید کیا سیکھنا ہے (What I Want to Explore More):

My Confidence Level (1-10): _______

🔄 Tomorrow's Preview

Tomorrow we'll learn about **Functions & Scope** where you'll build a **Utility Functions Library**!

You'll learn how to:

- Create reusable code blocks (functions)
- Pass data to functions (parameters)
- Return results from functions
- Understand variable scope (local vs global)

Get Ready By:

- ☐ Making sure your Cricket Counter works
- ☐ Thinking: What code did you repeat today?
- ☐ Imagine: How could you reuse code without copying?

📚 Resources (اگر مزید پڑھنا ہو)

Free Resources (3G-Friendly):

📖 MDN - Loops

- Link: https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Loops_and_iteration
- Best for: Understanding different loop types

🧑‍🏫 JavaScript Loops in Urdu

- Search: "JavaScript for loop while loop Urdu"
- Best for: Visual learners

💻 Practice Loops

- Try creating: countdown timers, multiplication tables, pattern printing

---

CodeSensei's Tip of the Day: 💡

"Loops are powerful but dangerous. Always ask yourself: 'Will this loop eventually STOP?' If you're not sure, add a safety counter (maxIterations). It's better to stop at 1000 iterations than freeze forever. Test with small numbers first (loop 5 times), then scale up!"

"لوپس طاقتور لیکن خطرناک ہیں۔ ہمیشہ پوچھیں: 'کیا یہ لوپ رُک جائے گا؟' اگر یقین نہیں تو safety counter لگائیں۔"

👥 Team Activity (Tomorrow's Standup)

Tomorrow at 7:00 PM, be ready to share:

1. Your Cricket Counter output (screenshot or copy-paste)
2. Highest score you got in simulation
3. One loop concept you mastered
4. One question about loops

کوڈ سیکھنا ایک سفر ہے، منزل نہیں۔ ہر دن ایک قدم آگے۔

"Learning to code is a journey, not a destination. One step forward every day."

Day 3 Complete! See you tomorrow for Functions! 🚀

اللہ حافظ! Tomorrow we make our code reusable! 🎯