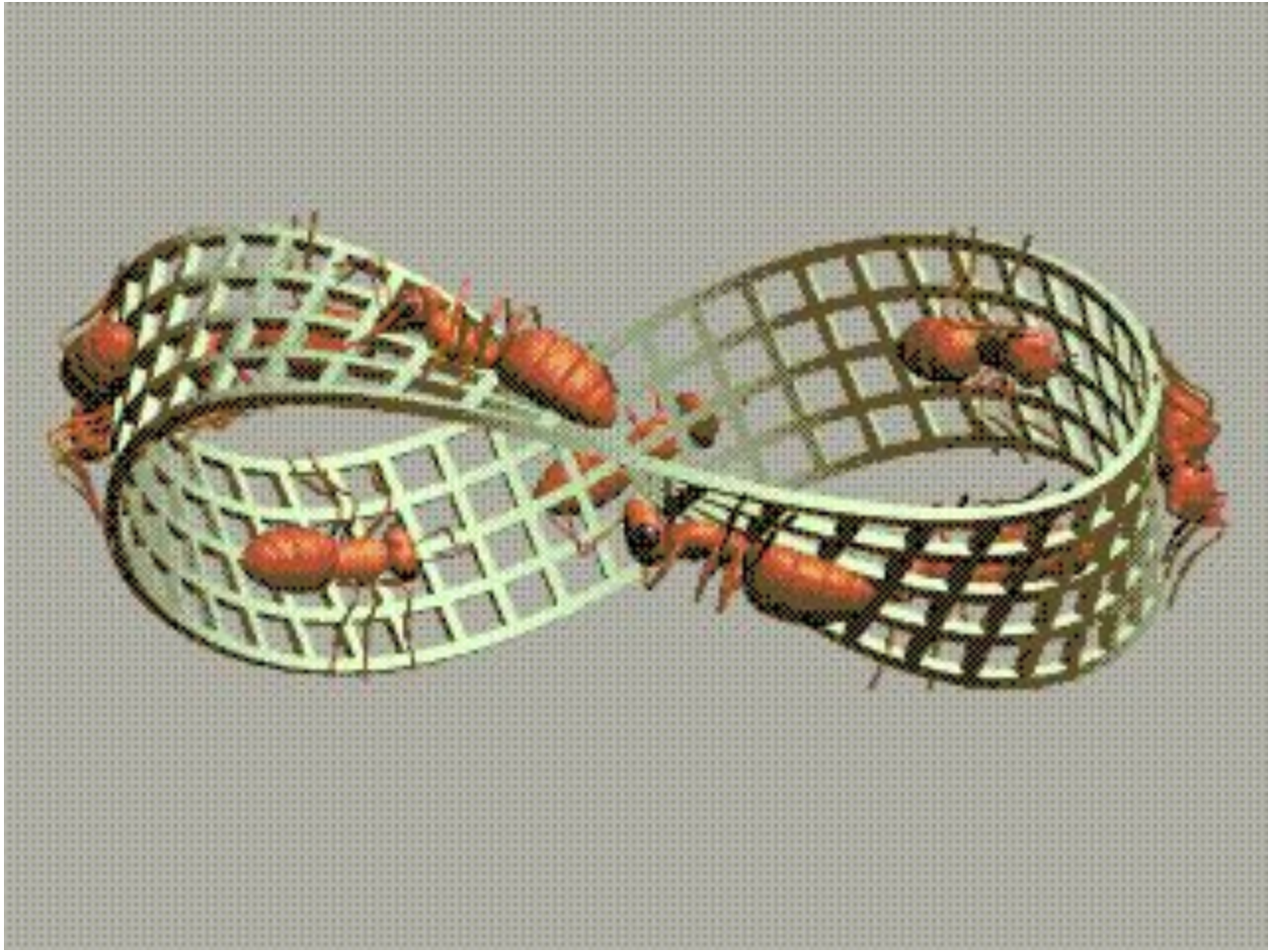
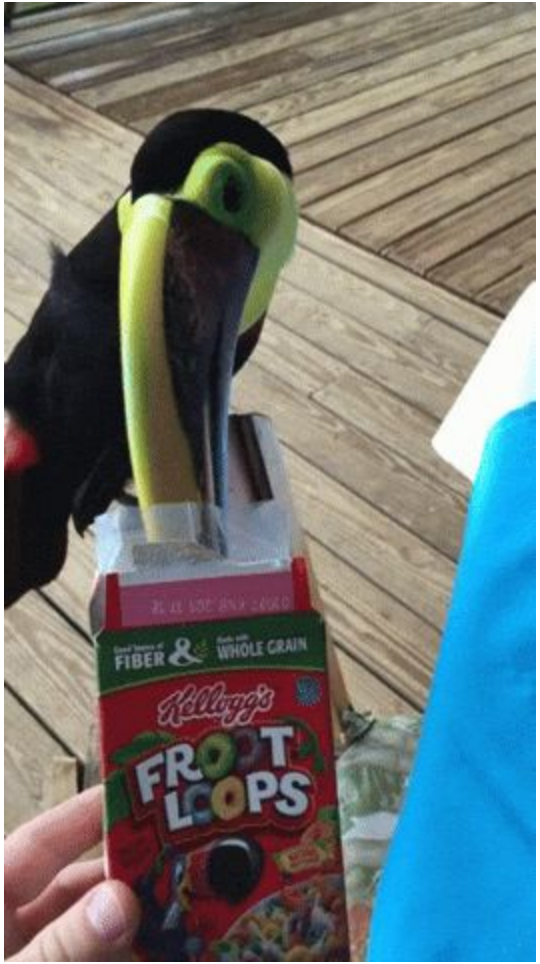
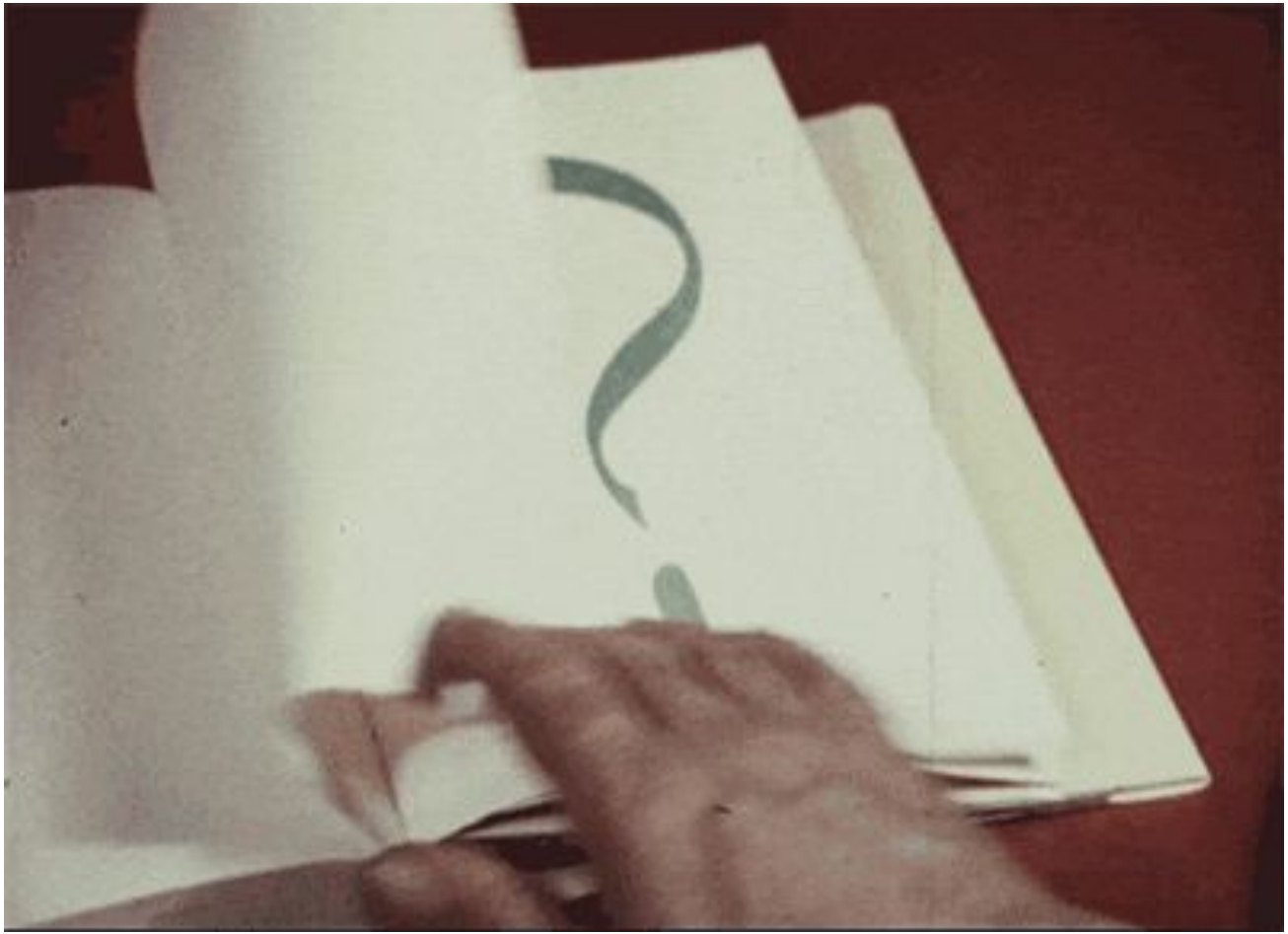
Lesson 4




FIBER &
WHOLE GRAIN
Kellogg's
FROOT LOOPS



Print 'C++ is powerful!' to the screen 100 times.


I WILL NOT DANCE ON
ANYONE'S GRAVE
I WILL NOT DANCE ON
ANYONE'S GRAVE
I WILL NOT DANCE
ANYONE'S GRAV
I WILL NOT DANC
* BELL RINGING *

You'd write the `cout` statement once... and then copy and paste it 99 more times. It works, but it's clumsy and inefficient.

There has to be a **better** way... **right**?

01
The Repetition Problem

02
Meet the Loops

03
The `while` Loop

04
The `for` Loop

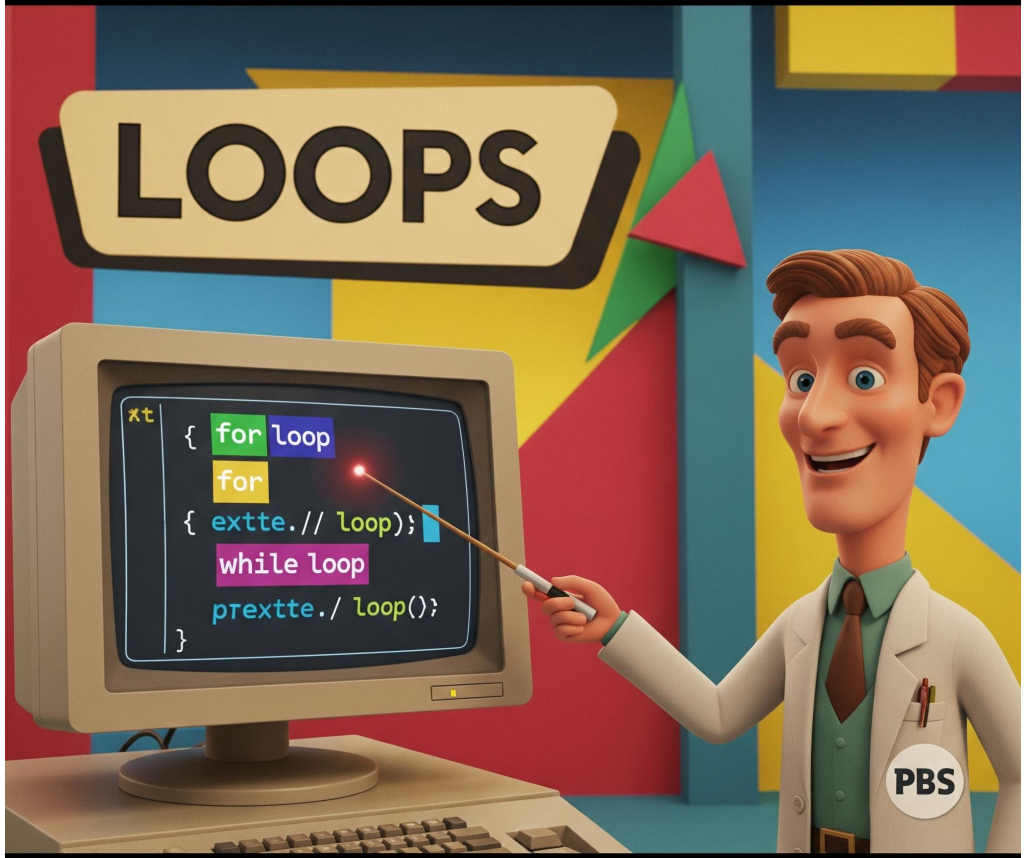
05
Loops in Action

06
Choosing Your Loop

1

Meet the Loops

The Elegant Solution


LOOPS
xt
{ for loop
for
{ extte.// loop);
while loop
prextte./ loop();
}
PBS

Loop

A piece of code that you would want to run in sequence multiple times.

Three Essentials of a Loop

- A starting point (Initialization)
- A condition to stop (Termination)
- A way to get from start to stop (Update)

2

The `while` Loop

Condition-Controlled Loop

```
while (condition) { //
code to repeat }
```
As long as the condition is true, the code inside the curly braces will execute.


Let's Count to 5!

Iteration	`start` Value	`start <= 5` is...	Output
1	1	True	1
2	2	True	2
3	3	True	3
4	4	True	4
5	5	True	5
6	6	False	(Loop stops)

Let's apply this to a classic problem: printing every number from 1 to 100. This is a perfect job for a `while` loop.

```
int start = 1;
while(start <= 100) {
cout << start << endl;
start++; }
```

This simple block does the work of 100 `cout` statements.

3

The `for` Loop

The Counting Loop

Anatomy of a `for` Loop

Initialization

Sets up the counter variable.

→

Condition

Checked before each iteration.

→

Update

Runs after each iteration.

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

All three essentials—initialization, condition, update—are on one line.

```
for(...) { for(...) {
...  }  }
```

You can put loops inside other loops to create complex patterns or work with 2D data like tables.

4

Loops in Action

Combining Our Tools

Print all numbers in a range that are both EVEN and multiples of 5. This requires a loop *and* a conditional `if` statement.

🎯 The `while` loop counts down, and the `if` statement filters the numbers, printing only those that meet both conditions.

5

Choosing Your Loop

The Right Tool for the Job

🤔

Use a `while` loop when you don't know the exact number of iterations, like waiting for user input.

🔢

Use a `for` loop when you know the exact count of iterations. It's a structured 'counting loop'.

> If your condition **never fails**, your loop will run forever and is called an **infinite loop**.

What happens if your condition to stop is never met?

Attendance

https://forms.gle/UB6UJMde8X2KZG7g8