For Loops Unpacked

For Loops must have:

1. A for statement, ending with a colon
2. A loop variable
3. An indented line of code

Example Code:

```
fruits = ["Apples", "Bananas", "Cherries"]

for x in fruits:
  print(x)
```

(**Here, the **x** is the loop variable**)

Output:

```
Apples
Bananas
Cherries
```

What is actually happening behind the scenes??:

Loop 1	With the **first** element in the dataset **in place of the loop variable** (here, **x** is the loop variable), the program completes the **full indented code** after the for statement	`for "Apples" in fruits:`
		`  print("Apples")`
		
		Output: Apples

Loop 2	With the **second** element in the dataset **in place of the loope variable,** the program starts the code again from the first line after the for statement, and completes the **full indented code** after the for statement	`for "bananas" in fruits:`
		`  print("Bananas")`
		
		Output: "Bananas"

Loop 3	With the **third** element in the dataset **in place of the loop variable,** the program starts the code again from the first line after the for statement, and completes the **full indented code** after the for statement	`for "Cherries" in fruits:`
		`  print("Cherries")`
		
		Output: "Cherries"

The loop will continue to restart with the next element in the dataset, until

1. it reaches the end of the dataset, or
2. a condition is met for the loop to stop.

A more complicated example:

```
fruits = ["apples", "bananas", "cherries"]

for fruit in fruits:
      if len(fruit) > 7:
            print(fruit[0])
```

Output:

```
B
C
```

Why?:

Loop 1	With the **first** element in the dataset **in place of the loop variable** (here, fruit is the loop variable), the program completes the **full indented code** after the for statement	`for "Apples" in fruits:`
		`  if len("Apples") > 7:`
		`  print("Apples"[0])`
		
		Output: Nothing. (The print statement is not executed, because the condition is not met)

Loop 2	With the **second** element in the dataset **in place of the loop variable,** the program starts the code again from the first line after the for statement, and completes the **full indented code** after the for statement	`for "Bananas" in fruits:`
		`  if len("Bananas") > 7:`
		`  print("Bananas"[0])`
		
		Output: B

Loop 3	With the **third** element in the dataset **in place of the loop variable,** the program starts the code again from the first line after the for statement, and completes the **full indented code** after the for statement	`for "Cherries" in fruits:`
		`  if len("Cherries") > 7:`
		`  print("Cherries"[0])`
		
		Output: C