Loops in Python :

Loops are used to execute a block of code repeatedly. The for loop is used for iterating over a sequence (like a list, tuple, or string) or other iterable objects.

The for Loop:

The for loop is ideal when you know how many times you want to repeat the code or when you want to process each item in a collection.

1. for loop with a sequence

This is the most common use of a for loop. It iterates directly over the items of a sequence, assigning each item to the loop variable one by one.

Syntax:

```
for item in sequence:
    # statements to execute
```

Example:

```
fruits = ["apple", "banana", "cherry"]
for fruit in fruits:
    print(fruit)
```

Output:

```
apple
banana
cherry
```

2. for loop with range()

When you want to loop a specific number of times, you use the range() function. This function generates a sequence of numbers, and the for loop iterates over them. This is where the concepts of initialization, condition, and increment are most clearly seen in a for loop.

Syntax for range():

Python

```
range(start, stop, step)
```

Connecting to Initialization, Condition, Increment:

Initialization: The start value of range() acts as the **initialization**. The loop variable starts at this value. If omitted, it defaults to 0.

Condition: The stop value provides the **condition**. The loop continues as long as the loop variable is less than the stop value.

Incrementation/Decrementation: The step value is the **increment** (or decrement, if negative). It's the value by which the loop variable is updated in each iteration. If omitted, it defaults to 1.

Example:

```
    # Loop from 2 up to (but not including) 10, in steps of 2.

    # start = 2  (Initialization)

    # stop = 10  (Condition: loop while i < 10)

    # step = 2   (Increment)


for i in range(2, 10, 2):

  print(i)
```

Output:

```
    2

    4

    6

    8
```