3. Program Flow Control

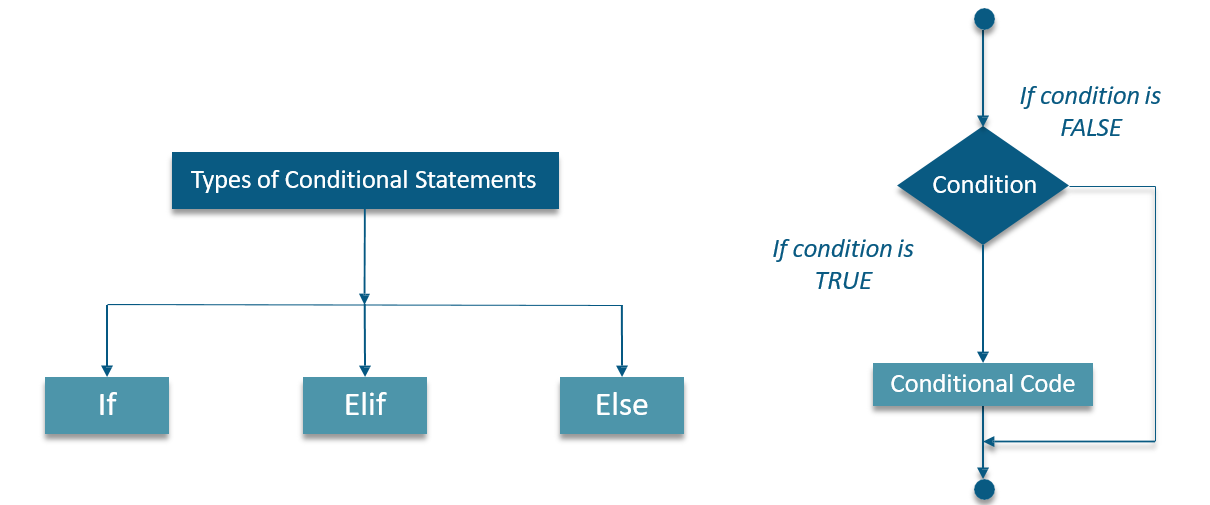
Table of Contents

|  |  |  |
| --- | --- | --- |
| Sl. No | Topics | Page No |
| 3.1 | **Conditional Statements** | **2** |
| 3.2 | **Loops & Control Statements** | **4** |
| 3.3 | **Command Line Parameters** | **9** |

## 3.1 Conditional Statements

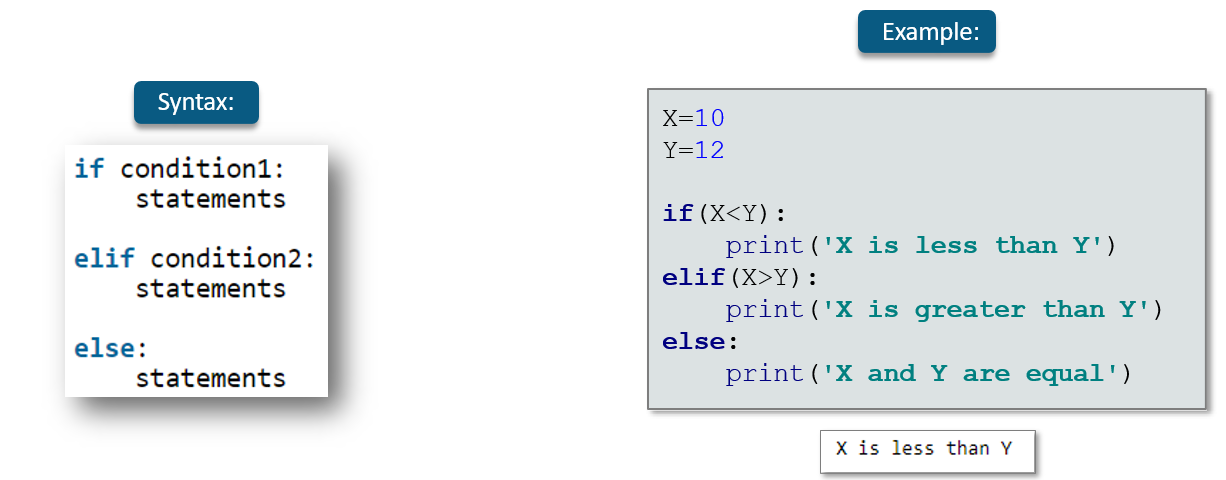
**What are Conditional Statements? What are its types?**

Conditional statements are used to execute a statement or a group of statements, when some condition is true.



**If Elif Else Statements**

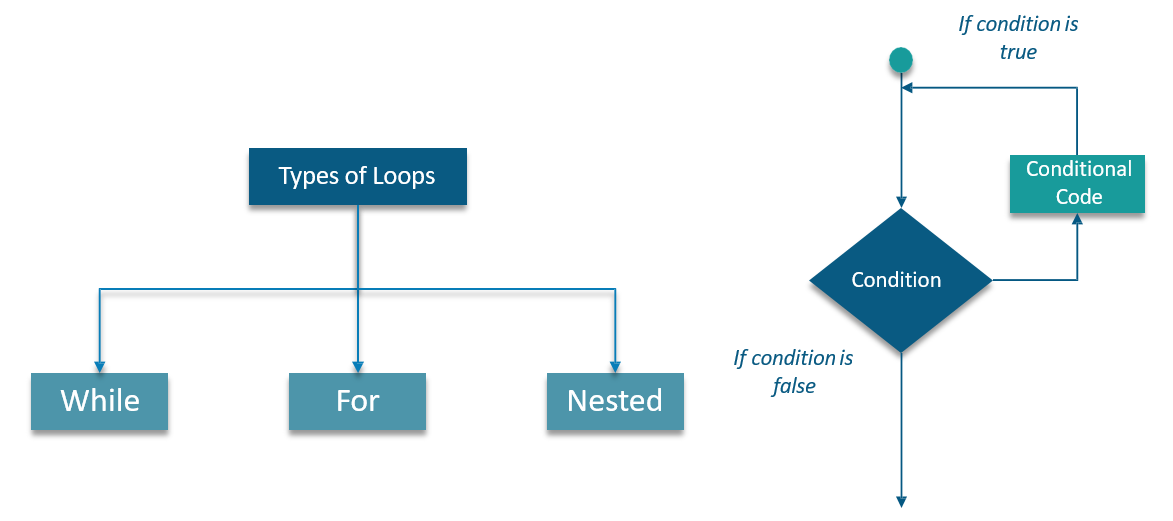
Consider the syntax & Example below:-



## 3.2 Loops & Control Statements

**What are Loops?**

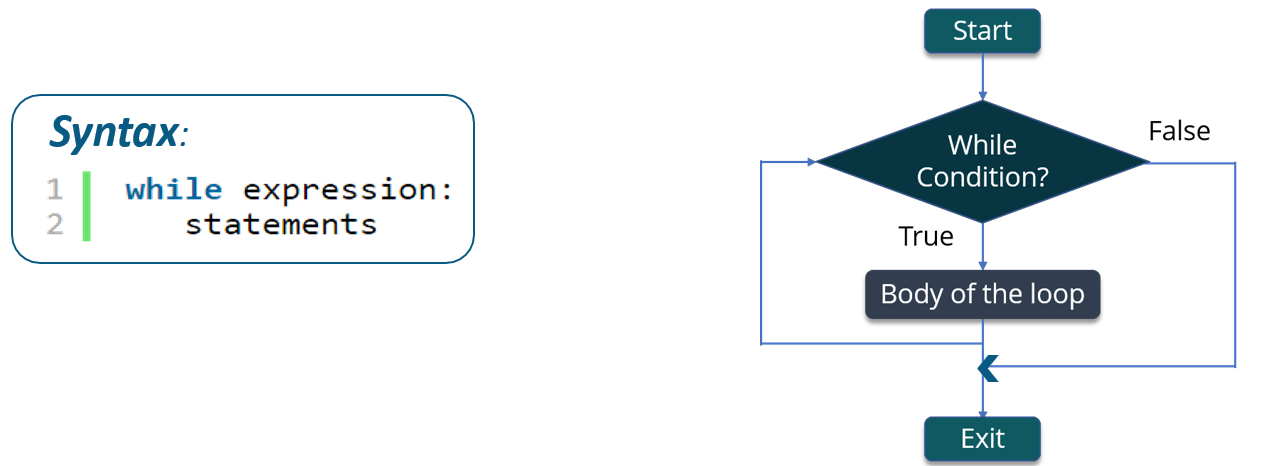
A loop statement allows us to execute a statement or a group of statement multiple times .



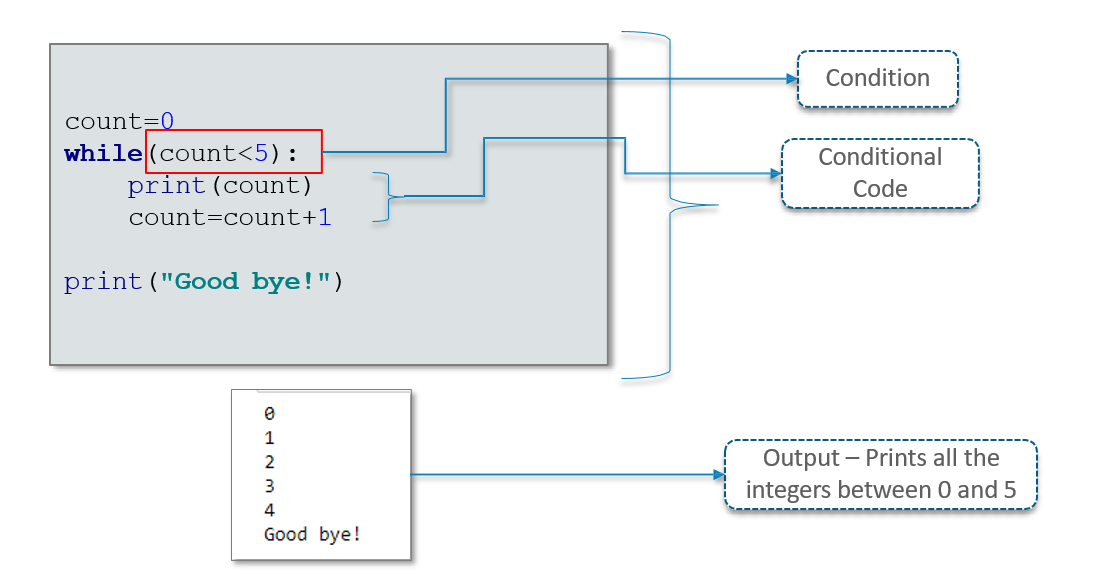
**While Loop**

“*While*” loops are known as indefinite or conditional loops. They will keep iterating until certain conditions are

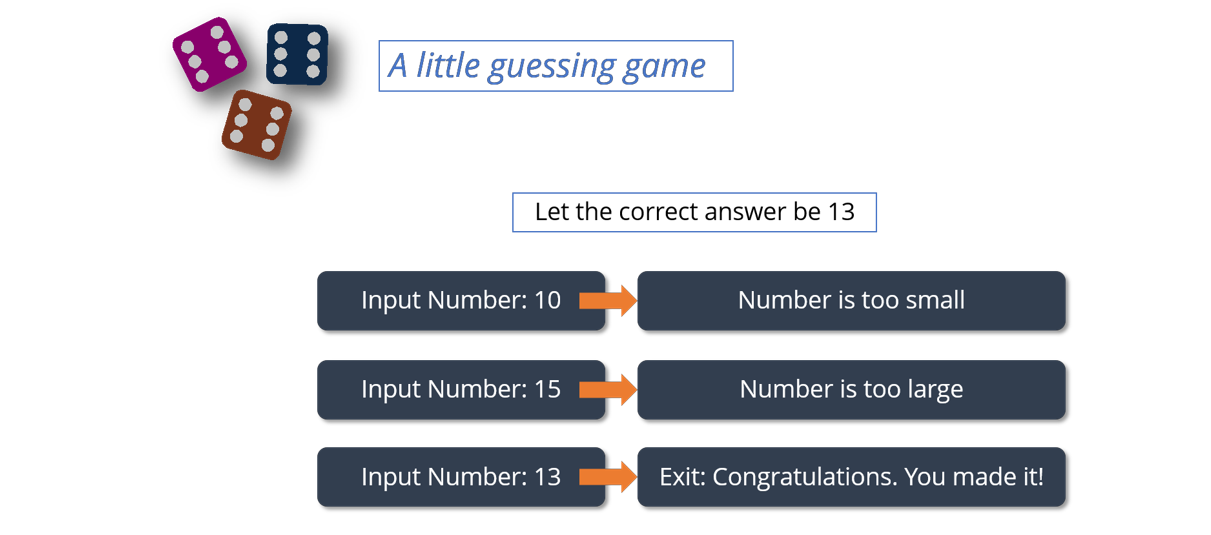
met. There is no guarantee ahead of time regarding how many times the loop will iterate.



**Example 1: While Loop**



**Example 2: While Loop**

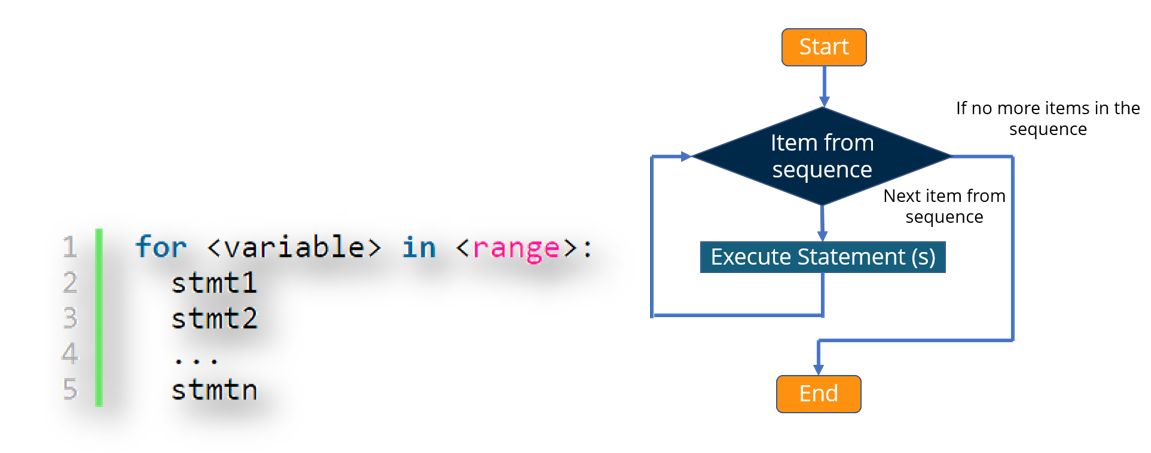


**For Loop**

“For” loop is a Python loop which repeats a group of statements a specified number of times.

The for loop provides a syntax where the following information is provided:

* + Boolean condition
  + The initial value of the counting variable
  + Incrementation of counting variable



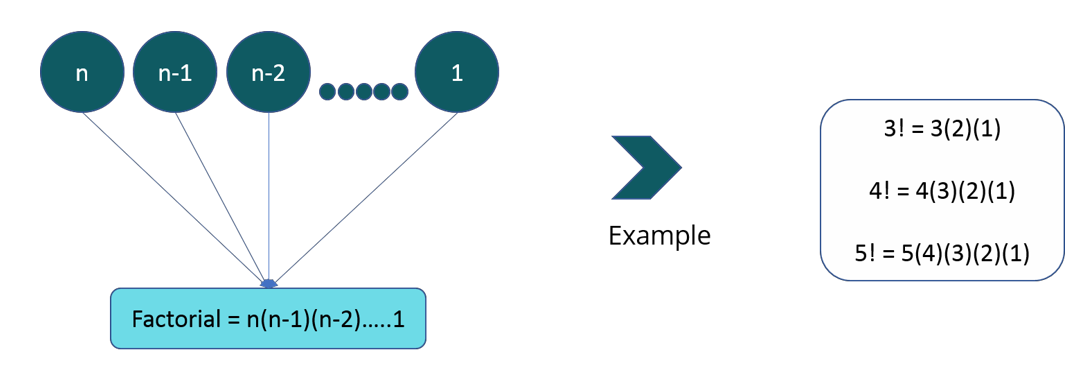
Difference between For and While loop is that, in While loop we don’t know the amount of iterations, where

as in For loop we are aware of how many times the block of code will be executed

fruits=[**'Banana'**,**'Apple'**,**'Grapes'**]

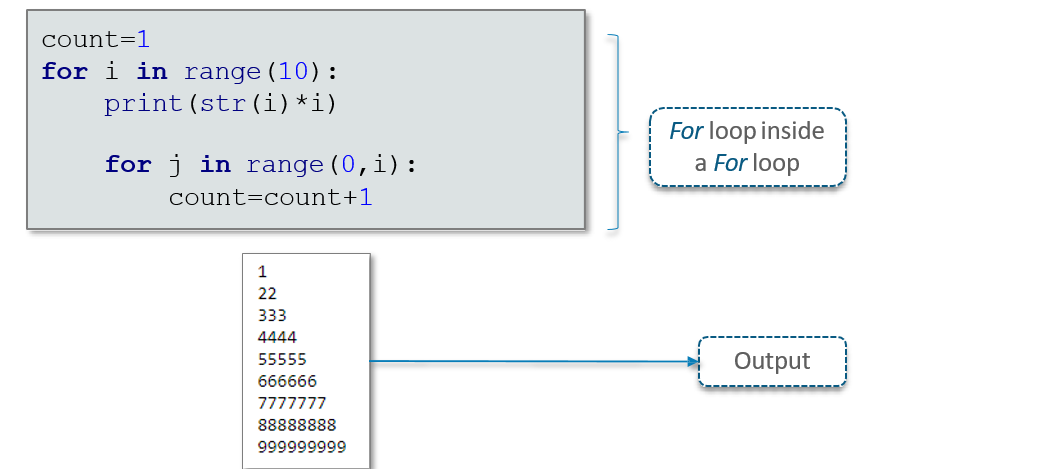
**for** index **in** range(len(fruits)): print(fruits[index])

We will be using “For” loop to write a program that calculates the factorial of any number



**Nested Loops**

Nested Loop, basically means a loop inside a loop. It can be a “For” loop inside a While loop and vice-versa. It can also be a While loop inside a While loop or “For” loop inside a “For” loop.



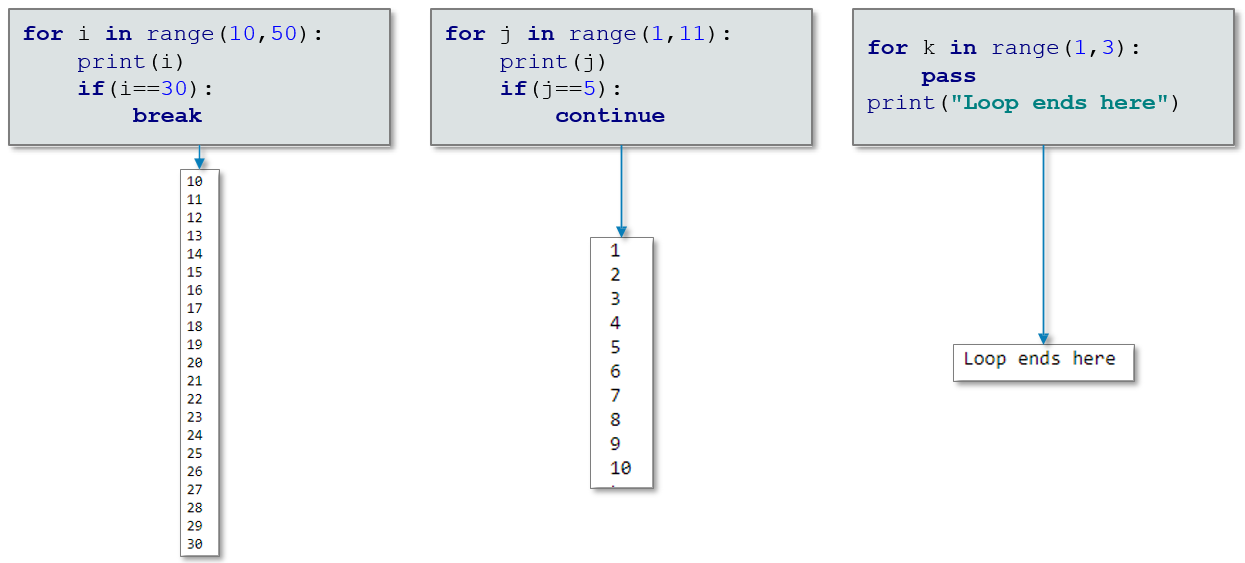
**Example: Nested Loops**

**What are Control Statements?**

Loop control statements change execution from its normal sequence. When execution leaves a scope, all automatic objects that were created in that scope are destroyed

|  |  |
| --- | --- |
| **Control Statement** | **Description** |
| break statement | Terminates the loop statement and transfers execution  to the statement immediately following the loop |
| continue statement | Causes the loop to skip the remainder of its body and immediately retest its condition prior to reiterating |
| pass statement | The pass statement in Python is used when a statement is required syntactically but you do not want any command or code to execute |

**Example: Control Statements**



## 3.3 Command Line Parameters

**What are Command Line Parameters?**

* It is possible to pass **arguments** to Python programs when they are executed
* The brackets which follow main are used for this purpose
* . argv refers to the number of **arguments** passed, and argv[] is a pointer array which points to each **argument** which is passed to main
* The Python **sys** module provides access to any command-line arguments via the **sys.argv**.

This serves two purposes:-

* + sys.argv is the list of command-line arguments
  + len(sys.argv) is the number of command-line arguments

**Example: Command Line Parameters**

