**For Loops**

In Python, a for loop is used to iterate over sequences like lists, strings, tuples, etc. Here’s a simple

Example:

languages = ['Swift', 'Python', 'Go']

for lang in languages:

print(lang)

Output:

Swift

Python

Go

Looping Through a String:

You can iterate over each character in a string using a for loop:

Example:

language = 'Python'

for char in language:

print(char)

Output:

P

y

t

h

o

n

Using range() with for Loop:

The range() function generates a sequence of numbers.range (start,end,step size) You can iterate over it like this

Example

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

print(i)

Output:

2 4 6 8

for Loop with else Clause:

You can add an optional else block after the for loop. It executes after the iteration completes

Example:

digits = [0, 1, 5]

for num in digits:

print(num)

else:

print("No items left.")

Output:

0

1

5

No items left.

Using for Loop for Repetition:

If you want to repeat an action without using list elements, you can use the \_ (underscore) as a placeholder variable

Example:

languages = ['Swift', 'Python', 'Go']

for \_ in languages:

print('Hi')

Output:

Hi

Hi

Hi

Nested for Loops:

You can have another for loop inside an outer loop:

Example:

for i in range(3):

for j in range(2):

print(f"Outer loop: {i}, Inner loop: {j}")

Output:

Outer loop: 0, Inner loop: 0

Outer loop: 0, Inner loop: 1

Outer loop: 1, Inner loop: 0

Outer loop: 1, Inner loop: 1

Outer loop: 2, Inner loop: 0

Outer loop: 2, Inner loop: 1

Zip() function:

It is used to combine elements from multiple iterables into tuple.Zip (iterable1,iterable2,……..)

Example

Names={“Sakshi ”,”ganesha”}

Roll=(21,05)

For name,roll in zip(name,roll):

Print(f”{name} have {roll} roll no)

Output

Sakshi have 21 roll no

Ganesha have 05 roll no

Enumerate()

Add a counter to an iterable and returns it as an object which can be looped over or converted to a list of tuple

Example:

Enumerate(iterable,start=0)

Fruits=[‘apple’,’banana’,’kiwi’]

For I,fruit in enumerate(fruits,start=1):

Print(i,fruit)

Output

1 apple

2 banana

3 kiwi

**While loop**

It repeateadly execute a block of (normaly indented ) statement as long as a test at the top keeps evaluating to a true value

We call it as a loop because control keep looping back to start of the statement until the test becomes false . Reading input until a condition is met

Syntax

While test: #loop test

Statement #loop body

Else: #optional else

Statement #run if didn’t exit loop with break

Example:

number = 1

while number <= 3:

print(number)

number += 1

output

1 2 3

* else Clause with while Loop:

A while loop can have an optional else clause that executes once the loop condition becomes false.

Run if and only if loop is exited normally

A while loop can have an else clause that is executed when the loop condition false

The else block is not executed if the loop is terminated by a break statement

Example:

counter = 0

while counter < 2:

print('This is inside loop')

counter += 1

else:

print('This is inside else block')

Output:

This is inside loop

This is inside loop

This is inside else block

* break Statement:

You can use break anywhere inside a while loop or for loop to terminate it immediately without checking the condition.

Example:

while True:

user\_input = input('Enter your name: ')

if user\_input == 'end':

print('The loop is ended')

break

print(f'Hi {user\_input}')

Output:

Enter your name: Kevin

Hi Kevin

Enter your name: end

The loop is ended

* Continue statement:

The continue statement in Python is a powerful tool for controlling loops. It jump to the top of the closet enclosing loop (to the loop headline)

It skips the current iteration and move to next one

Purpose of continue:

The continue statement allows you to skip the remaining code inside a loop for the current iteration.

It then jumps to the next iteration, without executing the rest of the loop body.

You can use it in both while and for loops.

Example

i = 0

while i < 10:

if i == 5:

i += 1

continue

print(i)

i += 1

Output: 0 1 2 3 4 6 7 8 9

Pass

Does nothing at all it is empty statement placeholder

The pass statement in Python is a null statement that serves as a placeholder

Example:

for i in range(10):

# Placeholder for future code

Pass