Content Covered

- Concept
- For Loop
- While Loop
- Comprehension
- Transfer Statement: break & continue

Loop

- Used to repeat code until the condition is met
- Used when we have pattern in code that can be repeated
- We can use for and while loop in Python

For Loop

Syntax

- for item in iterable:

  ---- statements ---

Example:

- for num in range(11):
  print(num ** 2)

- lis = [‘Nepal’, ‘US’, ‘China’]
  for country in lis:
  print(country.upper())

For Loop

```
for key, values in dict.items():
    print(f"{key} = {value}")


lis_1 = ['Python', 'Java']
lis_2 = [3, 4]
for course, dur in zip(lis_1, lis_2):
    print(f"Course: {course}, Duration: {dur}")
```

While Loop

Example – 1

```
a  = 5
while a <= 10:
    print(a)
     a = a + 1
```

Transfer

- Used to alter normal flow of loop
- **break** & **continue** are used for transfer
- break exits the loop where as continue skips one iteration
- Syntax:

```
while (condition):
    --- statements ---
    if (condition_2):
        break
```

For Else, While Else

- If loop doesn't exit due to normal condition, then else statement code is used
- Syntax:

```
while (condition):
    --- statements ---
    if (condition_2):
        break
else:
    -- statements --
```

Nested Loop

- When we have loop inside loop then we use nested loop
- Display multiplication table of 1-3.

Comprehension

- Short method to iterate over an iterable and generate new iterable
- square_list = [num**2 for num in range(5)]
- square_dict = { num:num**2 for num in range(5) }
- num_list = [1, 8, 10, 50, 79]
- print([num * 3 if num % 2 == 0 else num for num in num_list])

Exercise