04/07/2025

Day 4:

loops in python:

- used to repeat a block of code multiple times until a condition is met.

- loops helps in automating repeti-tive tasks.

1. for loop

2. while loop

3. Nested loop.

1. For loop:

- when you want to iterate over a sequence (list, tuple, string, etc).

Syntax:

for variable in sequence:

ex: 1. fruits = ["apple", "banana", 'cherry']

for fruit in fruits : print (fruit)

output: apple
banana
cherry.

2. using range () in for loop.

```
for i in range (1,6) :
        Print i
```

```
1
2
3
4
5
```

for loop, (:) next line, 4 spaces should be given.

indentation error. , colon missing: syntax error.

While loop:

A while loop runs as long as the condition is True.

for loop:

{dictionary . key: value pairs}

- keys ( )

  Print (person: keys ( ))

- values ( )

  Print (person. values ( ))

for i in range (1, 5)

1,
2
3
4

2

for i in range (5):

0
1
2
3
4

Control Statements:

1. break

~~2. skip~~

2. continue

~~4~~ 3. Pass

4. else.

1. break !

```
for i in range (1, 6):
    if i = = 3:
    break
    print (i)
```

output: 1,
2

enumerate : Positive indexing

```
0  1  2  3  4  5
M  a  h  e  s  h
-6 -5 -4 -3 -2 -1
```

negative indexing.

~~for index, value in enumerate (iterable~~

```
name = "Mahesh"
for index, char in enumerate (name):
print (index, char)
```

```
0 M
1 a
2 h
3 e
4 s
5 h
```

Nested for loops : loop inside another loop.

```
for i in range (2,6):
    for j in range (2,4):
        print (i, j)
```

Print a **pattern**:

```
for i in range (1,5):
    for j in range (i):
        print (i, end=" ")
```