Subject Artificial intelligence:

This notebook will teach you about “Exit from while loop, Iterations, while loop with else block, list comprehension, for loop” in the Python Programming Language. By the end of this lab, you'll know the basic concepts about “Exit from while loop, Iterations, while loop with else block, list comprehension, for loop”, and how to use these functions

Exit from the While Loop:

Use the break keyword to exit the while loop at some condition. Use the if condition to determine when to exit from the while loop, as shown below.

Example: Breaking while loop

```
num = 0

while num < 5:
   num += 1   # num += 1 is same as num = num + 1
   print('num = ', num)
   if num == 3: # condition before exiting a loop
      break
```

Output
num = 1
num = 2
num = 3

Continue Next Iteration:

Use the continue keyword to start the next iteration and skip the statements after the continue statement on some conditions, as shown below.

Example: Continue in while loop

```
num = 0

while num < 5:
        num += 1   # num += 1 is same as num = num + 1
        if num > 3: # condition before exiting a loop
                continue
        print('num = ', num)
```

Output
num = 1
num = 2
num = 3

While Loop with else Block:

The else block can follow the while loop. The else block will be executed when the Boolean expression of the while loop evaluates to False.

Use the continue keyword to start the next iteration and skip the statements after the continue statement on some conditions, as shown below.

Example: while loop with else block

```
num = 0

while num < 3:
        num += 1   # num += 1 is same as num = num + 1
        print('num = ', num)
else:
   print('else block executed')
```

Output
num = 1
num = 2
num = 3
else block executed

List Comprehensions:

Python makes it simple to generate a required list with a single line of code using list comprehensions

```
[27*x for x in range(1, 11)]
```

Output

[27, 54, 81, 108, 135, 162, 189, 216, 243, 270]

Note:

That's it! Only remember to enclose it in square brackets. There is possibility to make list comprehension conditional adding condition to syntax.

```
[27*x for x in range(1, 20) if x % 2 == 1]
```

Output

```
[27, 81, 135, 189, 243, 297, 351, 405, 459, 513]
```

For Loop

In Python, the for keyword provides a more comprehensive mechanism to constitute a loop. The for loop is used with sequence types such as list, tuple, set, range, etc.

The body of the for loop is executed for each member element in the sequence. Hence, it doesn't require explicit verification of a Boolean expression controlling the loop (as in the while loop).

Syntax:

```
for x in sequence:
   statement1
   statement2
   ...
   statement
```

To start with, a variable x in the for statement refers to the item at the 0 index in the sequence. The block of statements with increased uniform indent after the : symbol will be executed. A variable x now refers to the next item and repeats the body of the loop till the sequence is exhausted.

The following example demonstrates the for loop with the list object.

Example:

```
nums = [10, 20, 30, 40, 50]

for i in nums:
   print(i)
```

Output

```
10
20
30
40
50
```

For Loop with Tuple

```
nums = (10, 20, 30, 40, 50)
for i in nums:
    print(i)
```

Output

```
10
20
30
40
50
```

For Loop with String

```
for char in 'Hello':
    print (char)
```

Output

```
H
e
l
l
o
```

Reference

https://www.tutorialsteacher.com/

Questions

Stop the loop even if the while condition is true Exit the loop when i is 3?

Run a block of code once when the condition no longer is true with else statement?

print out a sequence of numbers from 0 to 7