Today I'll Cover :

1. Flow Control
2. Conditional Statements
3. Iterative Statements
4. Transfer Statements

Today I'll Cover :

1. Flow Control
2. Conditional Statements
3. Iterative Statements
4. Transfer Statements

Infinite Loop

A loop which keeps executing forever known as Infinite loop.

Nested Loop

A loop inside another loop is nothing but a nested loop.

Transfer Statement

1. **break** : We can use break statement inside loops to break loop execution based on some condition.
2. **continue** : We can use continue statement to skip current iteration and continue next iteration.

3. pass : In our programming syntactically if block is required which won't do anything then we can define that empty block with **pass keyword.**

pass

- It is an empty statement
- It is null statement
- It won't do anything

Loop with else block

- Inside loop execution, if break statement not executed, then only else part will be executed.

- else means loop without break

del

After using a variable, it is highly recommended to delete that variable if it is no longer required, so that the corresponding object is eligible for Garbage Collection. We can delete variable by using del keyword

Note:

We can delete variables which are pointing to immutable objects. But we cannot delete the elements present inside immutable object.

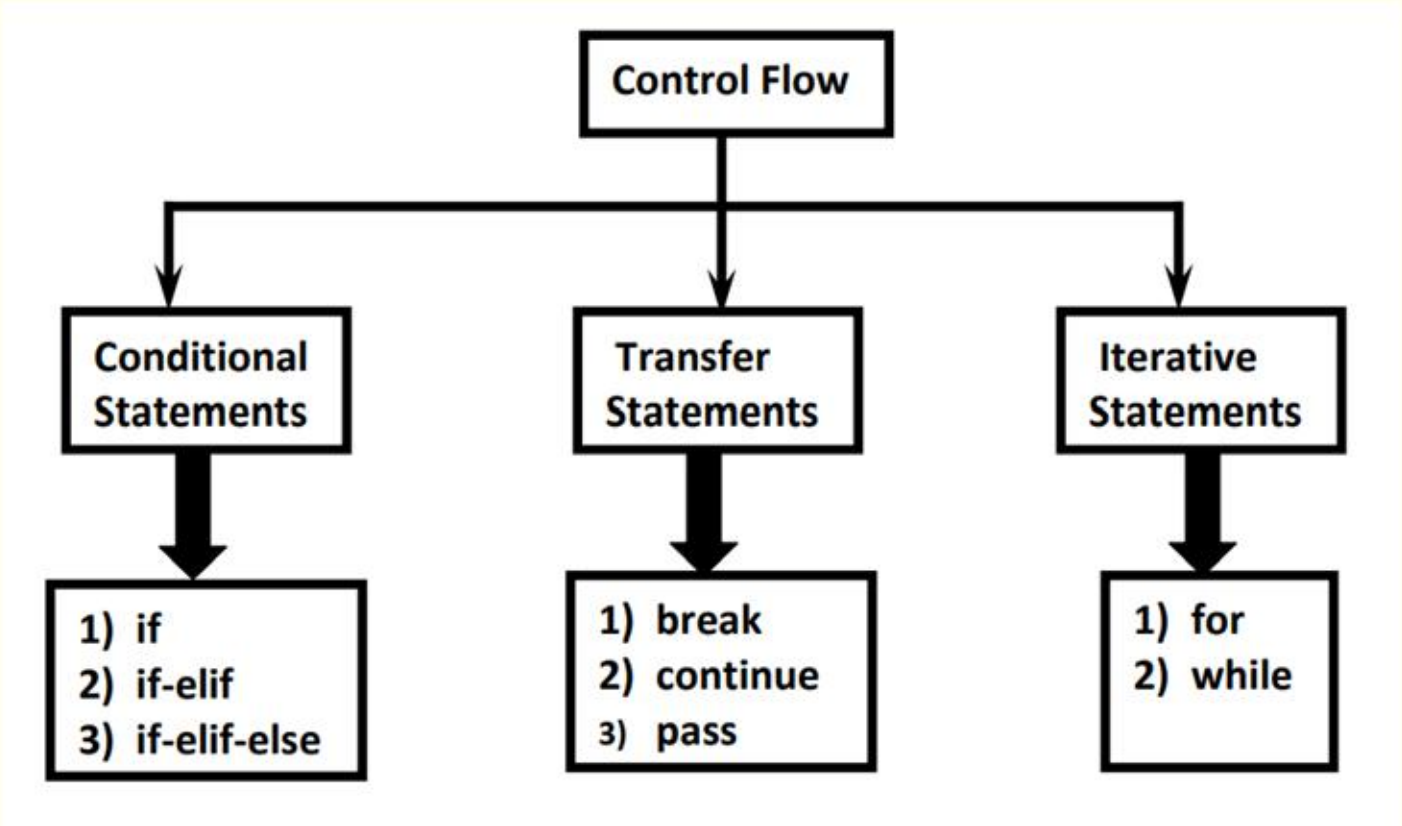
del	None
The variable will be removed and We can not access the variable .	The variable won't be removed but the corresponding object is eligible for Garbage Collection(re bind operation). Hence after assigning with None value, we can access that variable.

In the name of ALLAH,
the Most Beneficent, the Most Merciful

Mohammad Altaf Hussain

Flow Control

Flow control describes the order in which statements will be executed at runtime.



Conditional Statements (if)

Syntax (one line approach for single line):

```
if condtions : statement
```

Syntax (one line approach for Multiple line):

```
if conditons : statement1; statement
```

Syntax (Multiple line approach):

```
if conditions:
    statements
```

If condition is true then statement will be executed

Mohammad Altaf Hussain

Syntax (if-else):

```
if conditions:
    action1
else:
    action2
```

if condition is true then action1 will be executed otherwise action2 will be executed

Syntax (if-elif-else):

```
if conditions1:
    action1
elif condition2:
    action2
else:
    default action
```

Based condition the corresponding action will be executed.

Note: else is always optional.

Iterative Statement

If we want to execute a group of statements multiple times then we should go for Iterative statements.

Python supports 2 types of iterative statements.

1. for loop
2. while loop

For Loop

To execute some action for every element present sequence then we should go for ‘for’ loop.

Syntax:

```
for x in sequence :
    statements
```

While Loop

To execute a group of statements iteratively until some condition false, then we should go for while loop.

Syntax:

```
while condition :
    statements
```