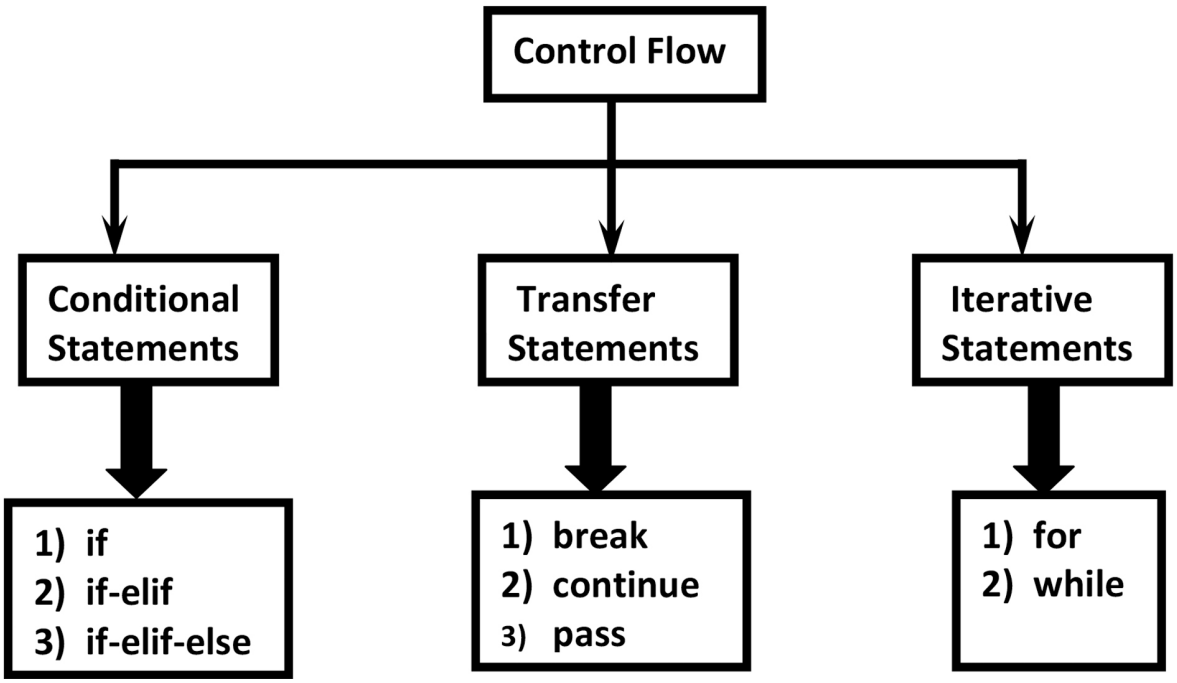
Flow Control

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



I. Conditional Statements

1) if

if condition : statement

or

if condition :
statement-1
statement-2
statement-3

If condition is true then statements will be executed.

2) if-else:

```
if condition :
    Action-1
else :
    Action-2
```

if condition is true then Action-1 will be executed otherwise Action-2 will be executed.

3) if-elif-else:

Syntax:

```
if condition1:
   Action-1
elif condition2:
   Action-2
elif condition3:
   Action-3
 elif condition4:
   Action-4
   ...
else:
   Default Action
```

II. Iterative Statements

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

1) for loop:

If we want to execute some action for every element present in some sequence(it may be string or collection)then we should go for for loop.

Syntax:

```
for x in sequence :
   body
```

2) while loop:

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

Syntax:

```
while condition :
    body
```

Nested Loops:

Sometimes we can take a loop inside another loop,which are also known as nested loops.

Eg:

```
1) for i in range(4):
2)     for j in range(4):
3)         print("i=",i,"   j=",j)
4)
```

III. Transfer Statements

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.

loops with else block:

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

else means loop without break

3) pass statement:

pass is a keyword in Python.

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**

del statement:

del is a keyword in Python.

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.