
DECODE
AI

Control Flow

→ Control flow refers to the order in which statements within a program executes.

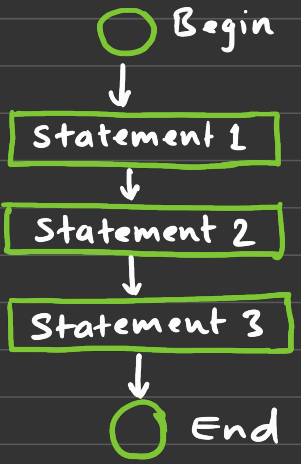
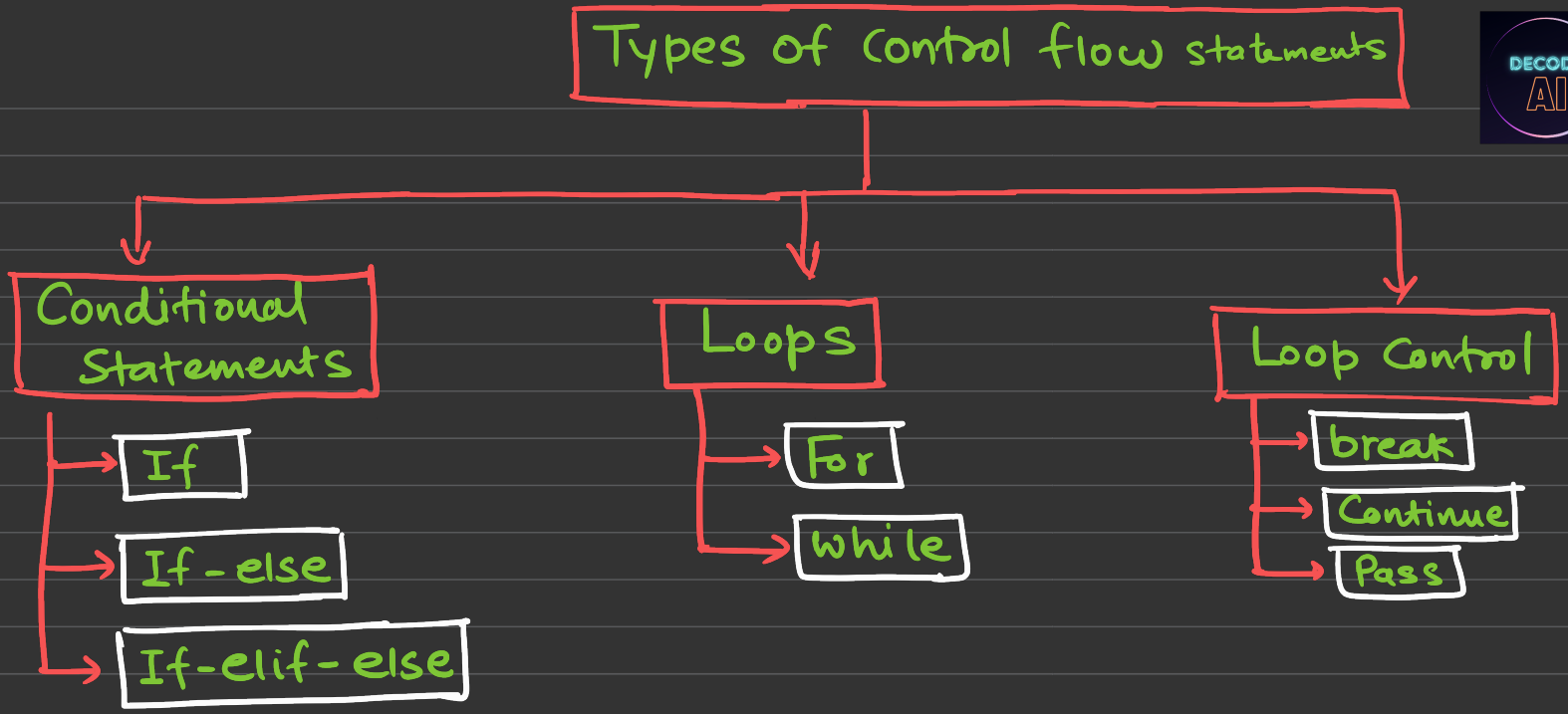


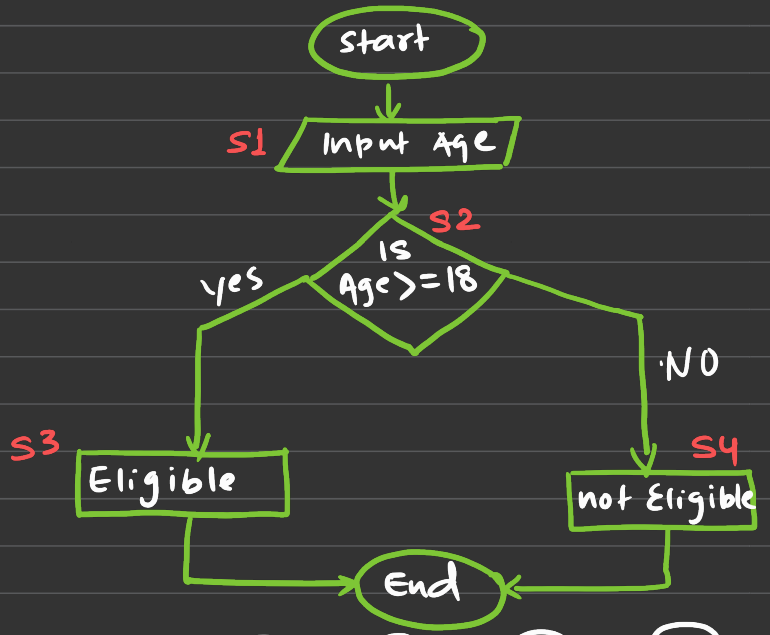
Fig: Sequential Control flow.

→ Control flow statements are fundamental components of programming languages that allow developer to control the order in which instructions are executed in a program.


Types of Control flow statements
DECODE AI
Conditional Statements
If
If-else
If-elif-else
Loops
For
while
Loop Control
break
Continue
Pass

Conditional statements

Example : Given the age of an user, the program should tell whether the user can vote or not?



Program (Pseudo Code)

S1: Input Age
S2: If Age >= 18
S3: Print - "Eligible"
S4: Print - "not Eligible"

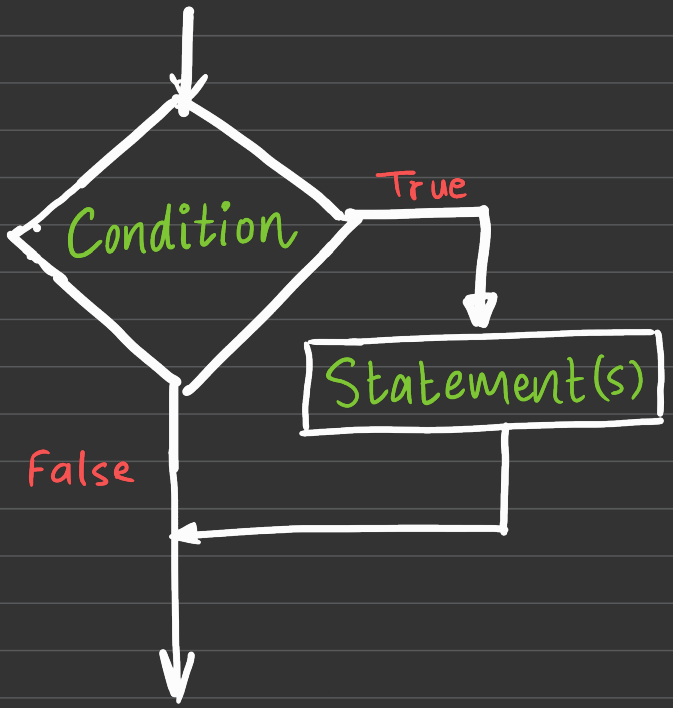
→ In Sequential flow : (S1) → (S2) → (S3) → (S4) → (End)

→ In Control flow

Age-50 : (S1) → (S2) → (S3) → (End)

Age-10 : (S1) → (S2) → (S4) → (End)

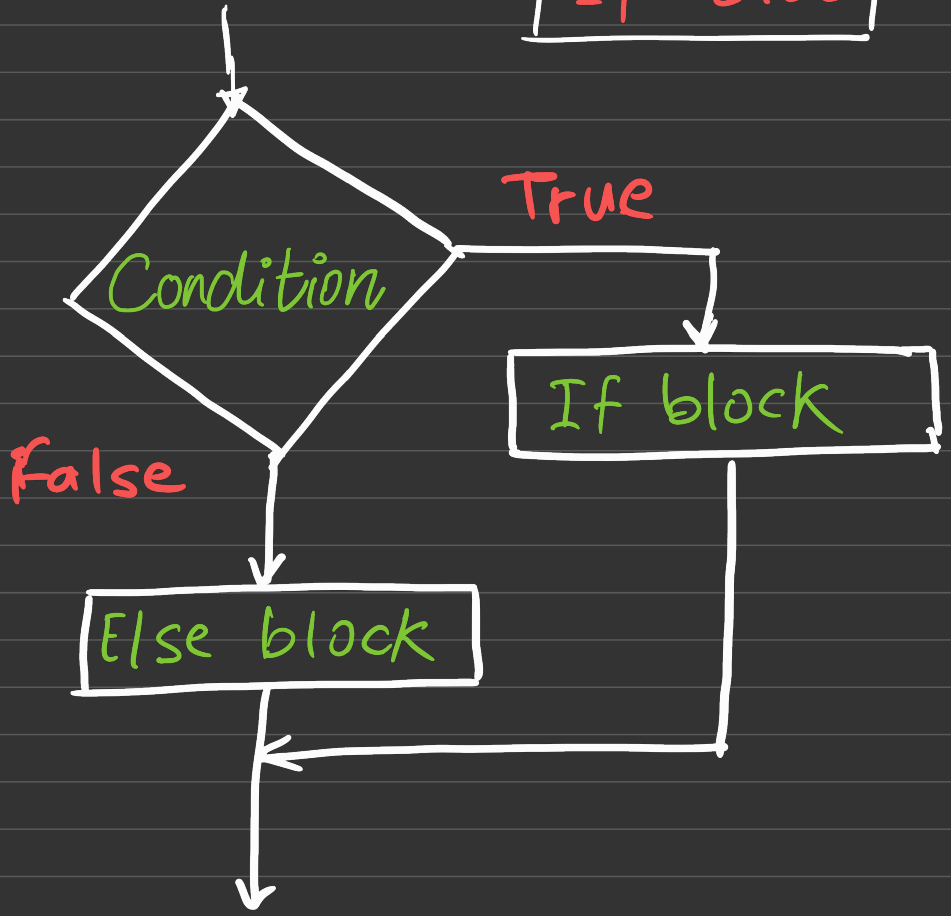
If



Pseudo Code

```
If condition:
    #if block statements
```

If-Else

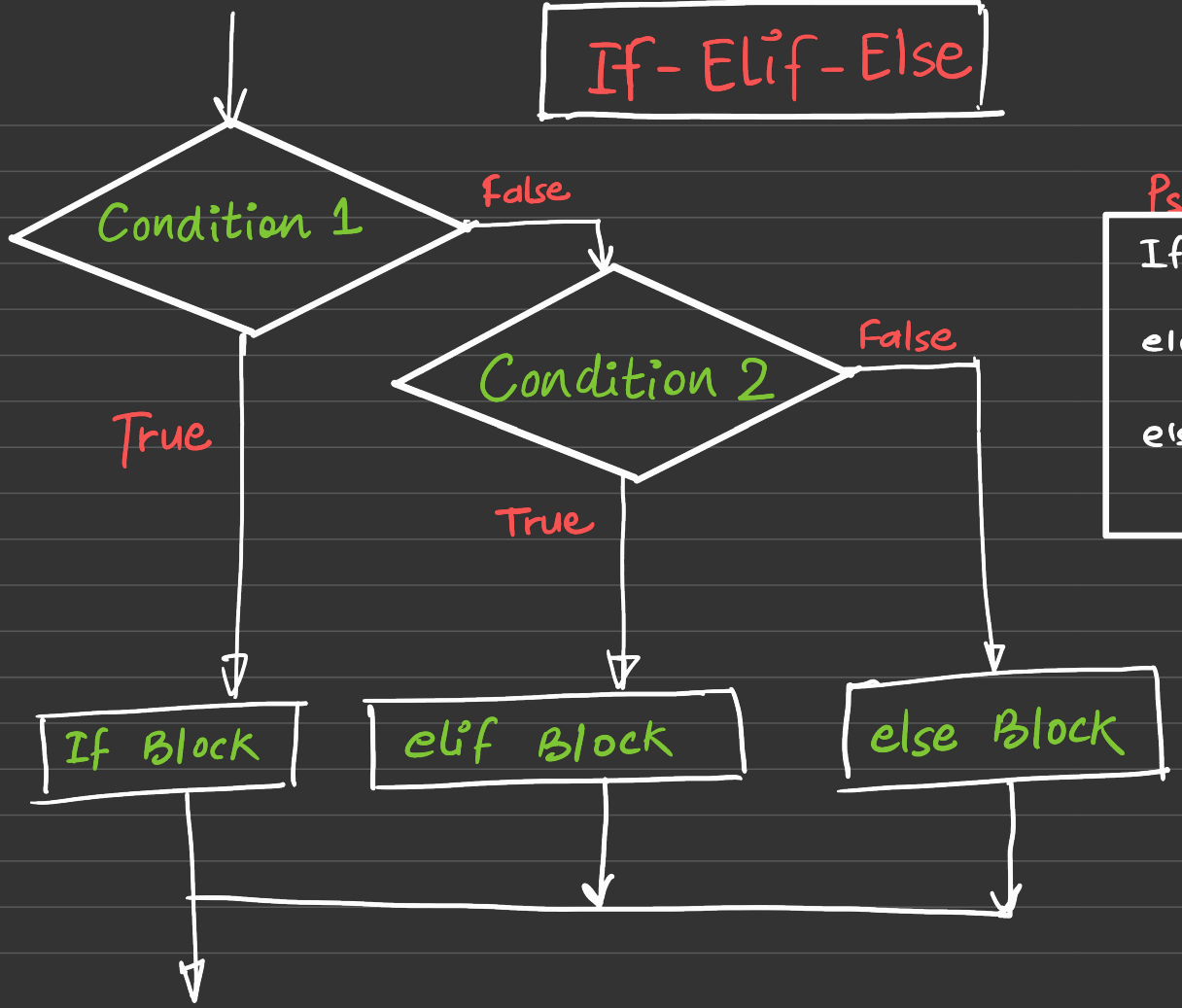


DECODE AI

Pseudo code

```
If condition:
    # if block statements
else:
    # else block statements
```

If-Elif-Else



Pseudo Code

```
If condition1 :
    #if block statement
elif condition2 :
    #elif block statement
else:
    #else block statement
```

Loops - For, while

→ A for loop purpose is to iterate over a sequence, be it a list, tuple, string or range and execute a designated block of code for each item in the sequence.

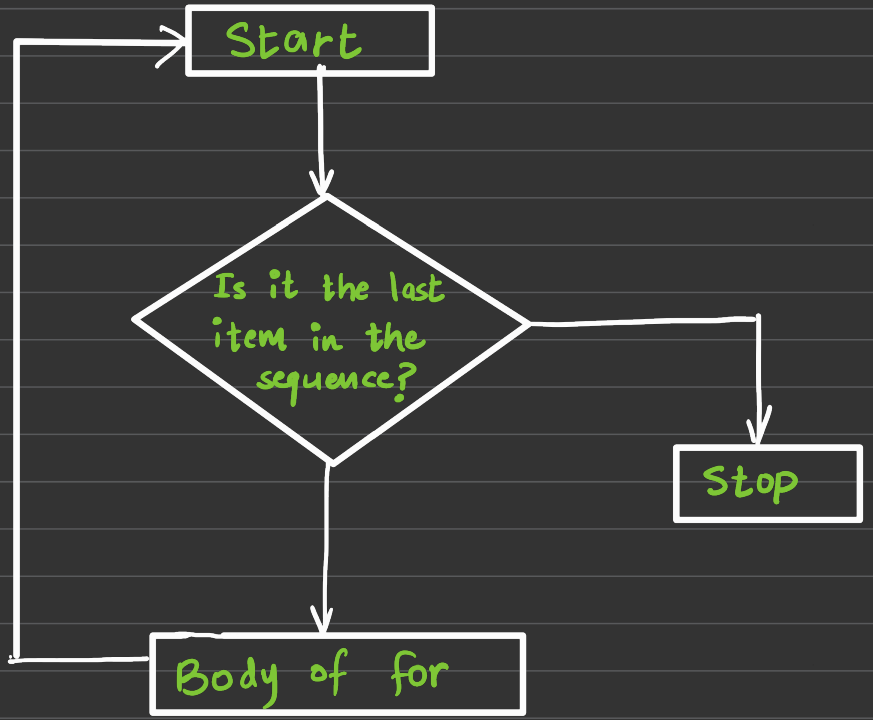
→ `range(start, stop, step)`

- → start: default 0
- → stop: iterate upto (stop-1)
- → step: Jump for each successive no.

Ex: `list(range(5, 50, 5))` ⇒ [5, 10, 15, 20, 25, 30, 35, 40, 45]

→ A while loop is used to iteratively execute a block of statements until a specified condition is met.

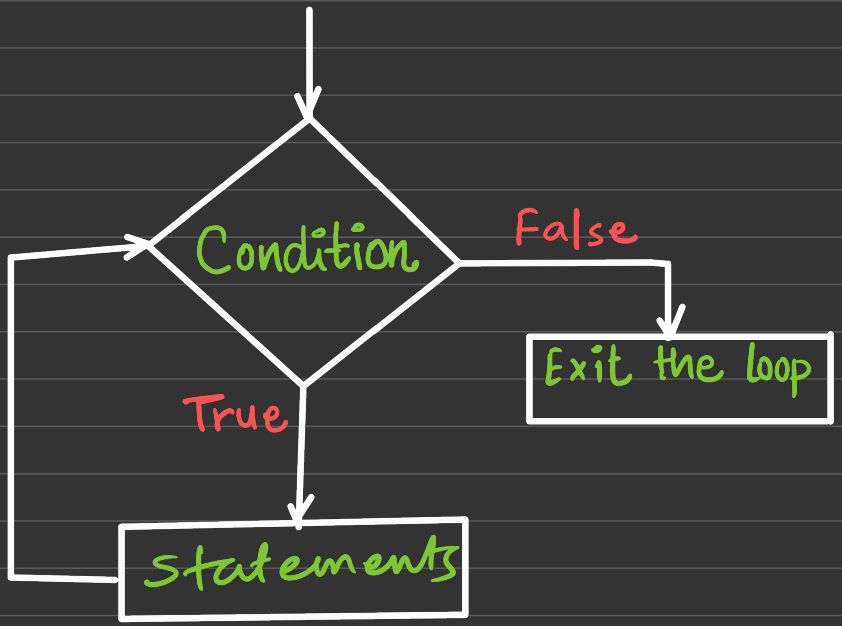
For



Pseudo code

```
for i in range(5):
    print (i)
```

While



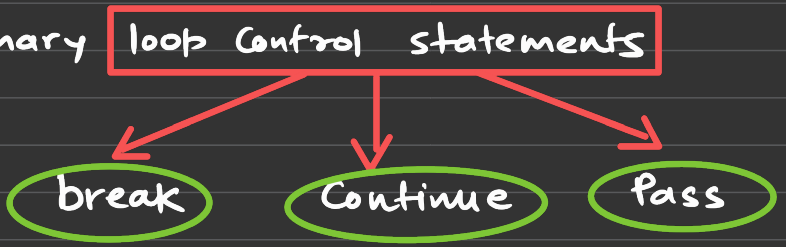
Pseudo code

```
count = 0
while count < 5:
    print(count)
    count += 1
```

Loop Control

→ In python loop control statements change the flow of execution in loops (for or while).

→ There are three primary loop control statements



* **Break**

→ Exit the loop prematurely when a condition is met.

```
for i in items:
    if i==x:
        break
    print(i)
```

* **Continue** ✓

→ skip the rest of current iteration and move to next one.

```
for i in items :
    if i%2==0:
        continue
    sum += i #sum of odd no.
```

* **Pass**

→ no operation placeholder

→ when you are writing structure of your code and plan to implement it later

→ during debugging.