KUMARAGURU college of technology
character is life

KCED
Learning Augmented

Looping/Iteration Constructs

KUMARAGURU college of technology
character is life

KCED
Learning Augmented

LEARNING AUGMENTED

Overview of this Lecture

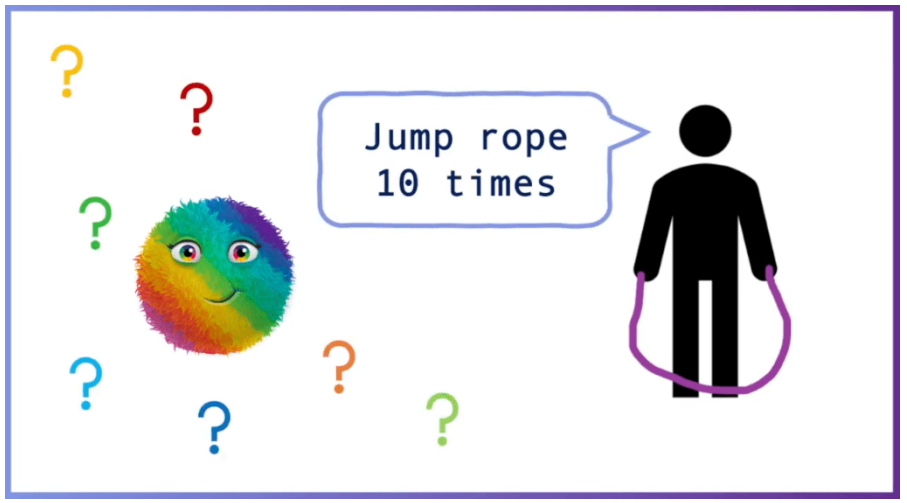
- **Iteration**
  - **While**
  - **For**
  - **Break**
  - **Continue**
  - **Pass**

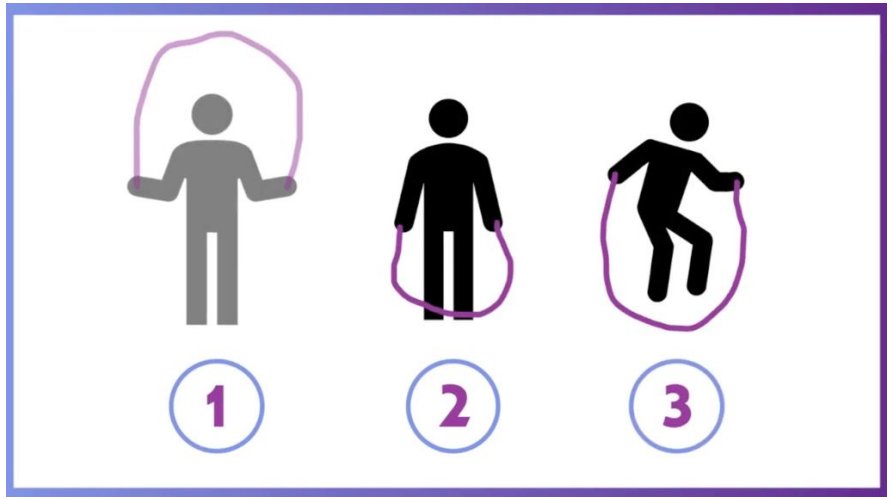
KUMARAGURU college of technology
character is life

KCED
Learning Augmented

LEARNING AUGMENTED

What is looping?





Introduction

- Looping constructs are **for** and **while**.
- The conditional statements and looping constructs are the basic building blocks of any programming language.
- Control statements or looping constructs are used to repeatedly execute a block of code in a program.
- There is usually a **counter** associated with the loop that indicates how many times the block of code is to be executed, or a **condition** that is checked every time to see if the block of code should be executed again.

Types of looping constructs

Python provides two types of looping constructs:

- while loop
- for loop

- There are two major kinds of loop programming:
  - Event-controlled/Condition-controlled loops
  - Count-controlled loops

Kinds of loops

Event-controlled loop: / Indefinite loop

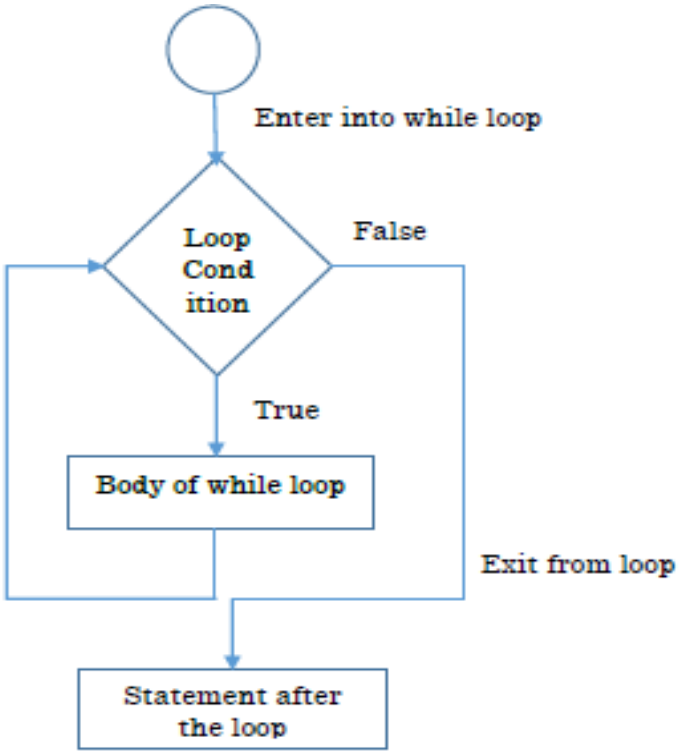
- The programmer stops the loop execution when a condition is no longer true.
- In Python, while statement shall be used for this – it executes the loop body while the condition is true.
- The while statement checks the condition before performing each iteration of the loop.

Count-controlled loop / Definite loop

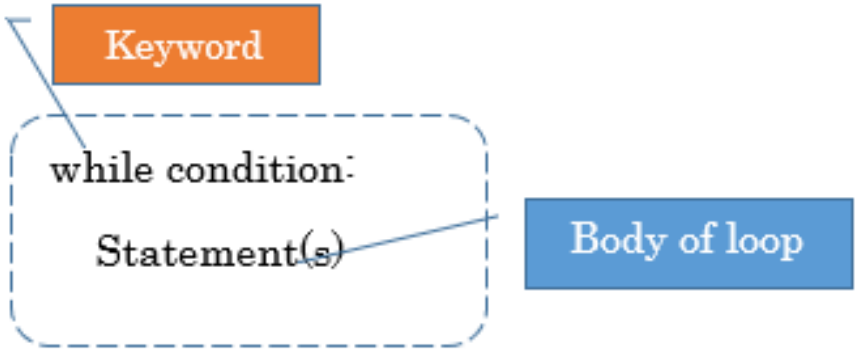
- **T**he programmer knows at the beginning of the loop execution how many times it needs to execute the loop.
- In Python, this kind of loop is defined with **for** statement, which executes the loop body for specific number of times.
- Counting loops are subset of event-control loop, where the loop is repeated until the required number of iterations is reached.

while loop

- The while loop in Python is used to iterate over a block of code as long as the condition remains true.
- While loop is a **condition controlled** loop.



Syntax:



About while loop:

- ✓ Condition is a Boolean expression which produces True or False result on upon evaluation.
- ✓ A colon(:) must be followed by the condition.
- ✓ Block of code may contain one or more statements.

But there is no hard and fast rule to be followed to always use while loop for event controlled loop, it can also be used for count-controlled loop. Thus, based on the scenario, the programmer has the freedom to choose the appropriate looping constructs for implementation.

Points to remember:

- In if statement, block of code is executed only once if the condition is true, else the block of code never gets executed.
- But in while construct, block of code will be executed repeatedly until the condition becomes false i.e the construct checks the condition again and again, executes the block of code as long as condition evaluates to be true.
- You may think that, when does this repeated execution gets terminated? it will be terminated when the condition becomes false.
- This kind of repeated action is called as loop. This process is also known as iterating over a block of code or iteration.
- While loop may not execute even once, if the condition evaluates to false, as initially the condition is tested before entering the loop.

Example:

- Program

```
#loop executes until the user respones as "no"

answer="yes"              # initialization
while(answer != "no"):    # condition
   print("Welcome to loop world!!!")   # print greet message
   answer=input("Say, yes/no:")
```

- Output:

```
Welcome to loop world!!!
Say, yes/no:yes
Welcome to loop world!!!
Say, yes/no:yes
Welcome to loop world!!!
Say, yes/no:yes
Welcome to loop world!!!
Say, yes/no:no
```

KUMARAGURU college of technology
character is life

KCED
Learning Augmented

LEARNING AUGMENTED

Write a program to print the sentence “Python is the most powerful programming language” for 5 times.

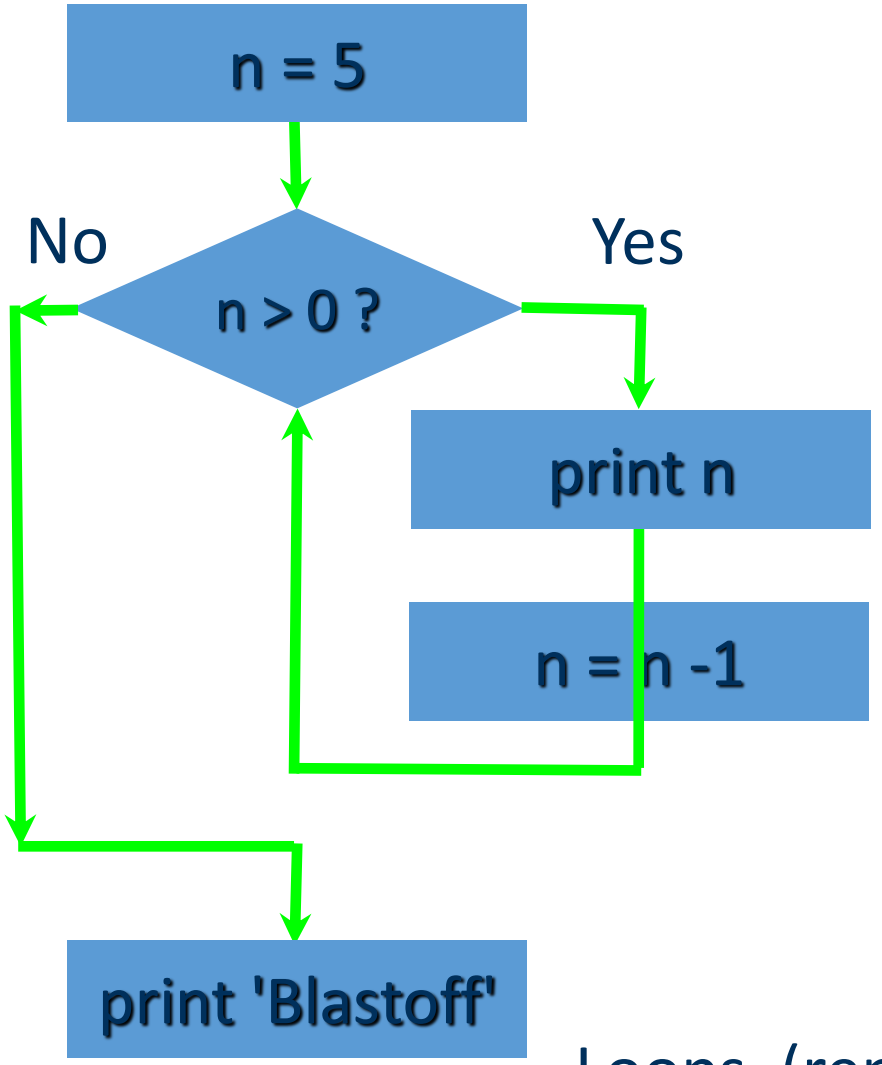
Without loop	Using loop

#To print the sentence 5 times
quote="Python is the most powerful programming language"
print(quote)
print(quote)
print(quote)
print(quote)
print(quote)
	#To print the sentence 5 times
	quote="Python is the most powerful programming language"
	count=1 # loop variable initialization
	while(count<=5): # loop until it reaches 5
	print(quote) # prints the value of quote
	count=count+1 # loop variable update

KUMARAGURU college of technology
character is life

KCED
Learning Augmented

LEARNING AUGMENTED



Repeated Steps

Program:

```
n = 5
while n > 0 :
    print(n)
    n = n -1
print('Blastoff!')
print(n)
```

Output:

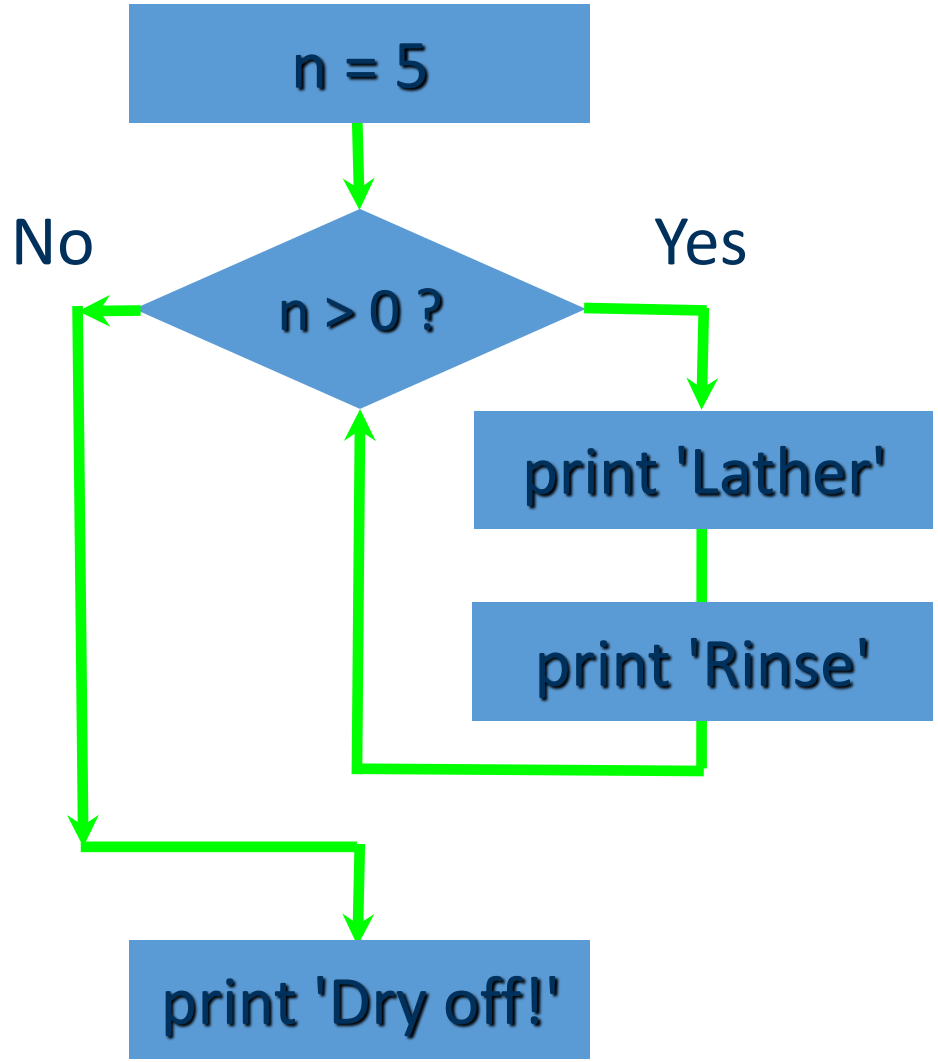
```
5
4
3
2
1
Blastoff!
0
```

Loops (repeated steps) have iteration variables that change each time through a loop. Often these iteration variables go through a sequence of numbers.

KUMARAGURU college of technology
character is life

KCED
Learning Augmented

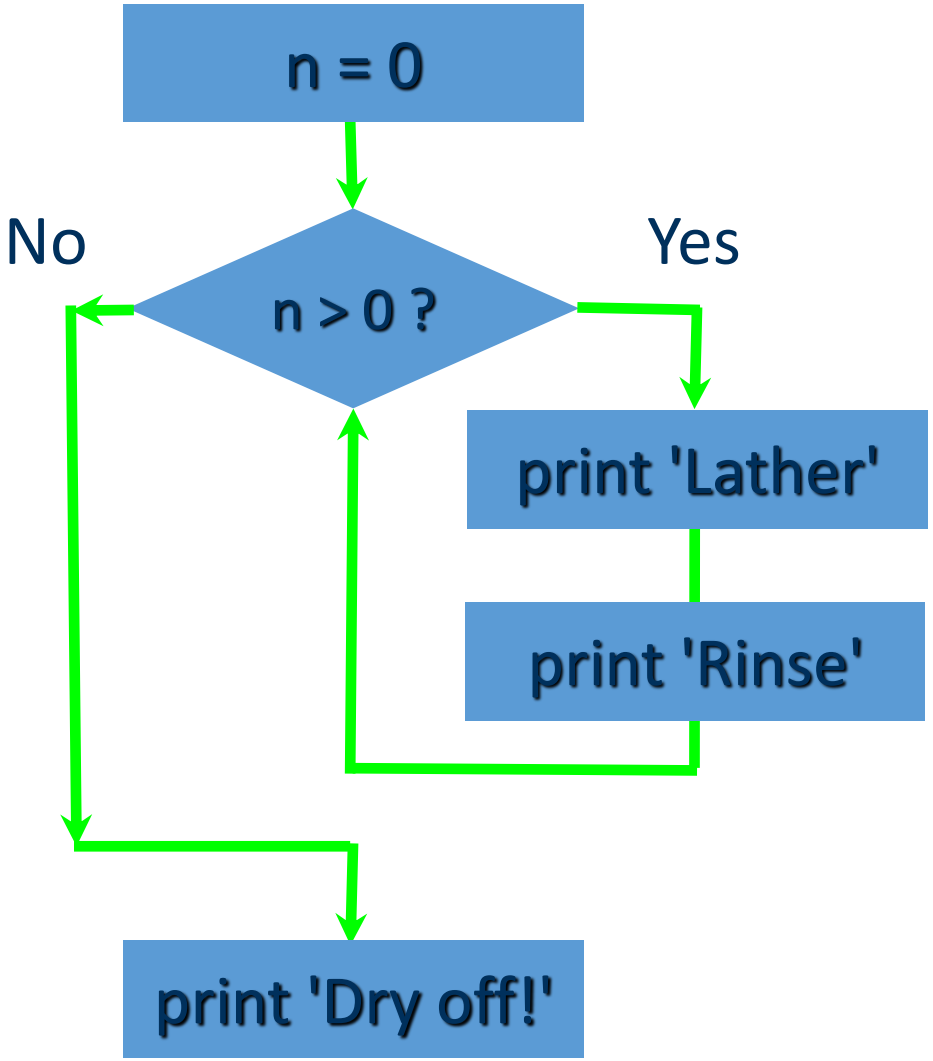
LEARNING AUGMENTED



An Infinite Loop

```
n = 5
while n > 0 :
    print('Lather')
    print('Rinse')
print('Dry off!')
```

What is wrong with this loop?



What does this loop do?

```
n = 0
while n > 0 :
    print('Lather')
    print('Rinse')
print('Dry off!')
```

KUMARAGURU college of technology
character is life

KCED
Learning Augmented

Write a program to print the numbers from 1 to 10

```
>>> c=1
>>> while(c<=10):
        print(c,end=" ")
        c=c+1
```

```
1 2 3 4 5 6 7 8 9 10
```

LEARNING AUGMENTED

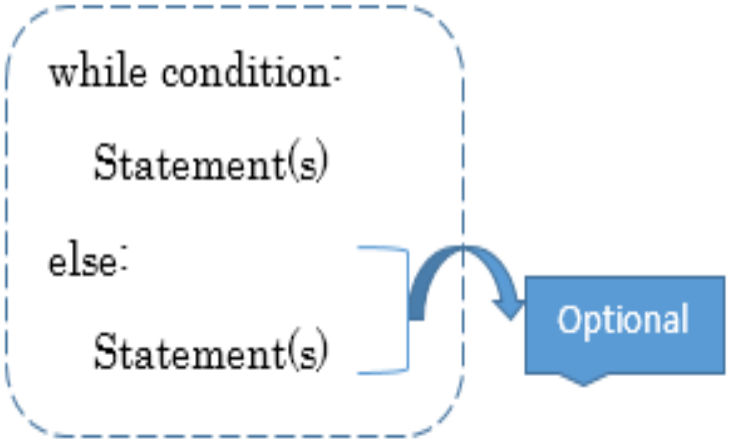
KUMARAGURU college of technology
character is life

KCED
Learning Augmented

while loop with else

- Python allows to use else statement associated with a loop statement.
- The while() loop can have an **optional** else block.
- The else part is executed if the condition in the while loop evaluates to false.
- While loop shall also be terminated with a break statement
- In such case, the else part is ignored.
- Hence, a while loop's else part executes if no break occurs and the condition evaluates to false.
- The programming languages such as C, C++, etc., doesn't have loop with else part feature.

Syntax:



KUMARAGURU college of technology
character is life

KCED
Learning Augmented

LEARNING AUGMENTED

Example: while loop else part

```
n = 0
while n < 5:
  print(n, " is  less than 5")
  n = n + 1
else:          # executes when condition n<5 evalutes to false
  print(n, " is not less than 5")
```

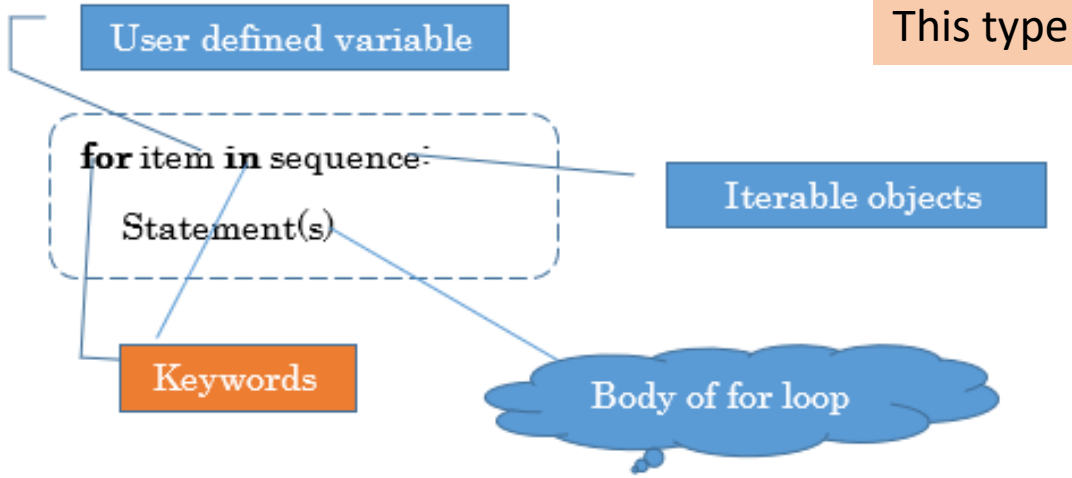
- Output

```
0  is  less than 5
1  is  less than 5
2  is  less than 5
3  is  less than 5
4  is  less than 5
5  is not less than 5
```

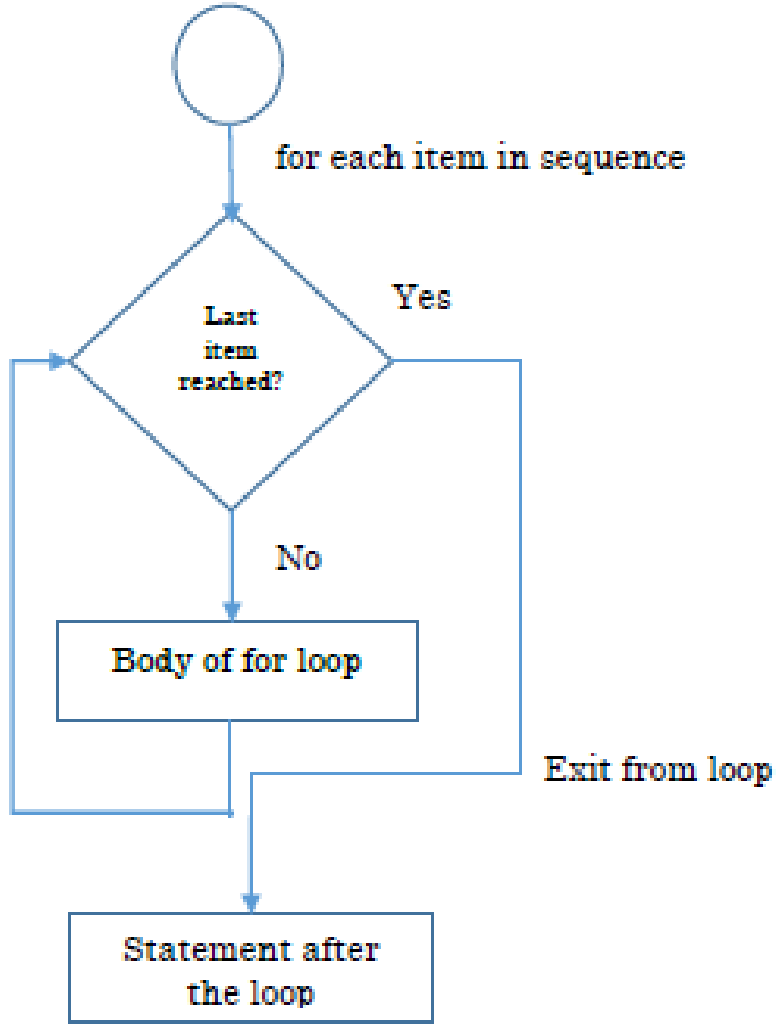
for loop

- The **for** loop in Python is slightly different from for loop in other programming language.
- The **for** loop is used to iterate over a sequence of objects such as list and string.
- It iterates through each item in a sequence, where the sequence of object holds multiple data item.
- The term “iterable objects” and “sequence” refers to the type objects, such as string, list, tuples and dictionaries.
- The for loop can be written in two formats:

for loop: Type 1:



This type of **for** loop iterates over any iterable objects.



- **for** is a keyword which begins the loop.
- This loop will be executed until it reaches the end of the sequence.
- Note that, there is no explicit condition to terminate the loop.
- For example, if the sequence has 5 items, then the loop will be executed 5 times.
- The keyword **in** is used to iterate over a sequence.
- **item** is the user defined loop variable that takes the value of the item inside the sequence on each iteration.
- The statements in the body of the loop will be executed once for each time.

KUMARAGURU college of technology
character is life

KCED
Learning Augmented

LEARNING AUGMENTED

Example

- Consider a string ”Python”, write python code to print each character of a string one by one using for loop.

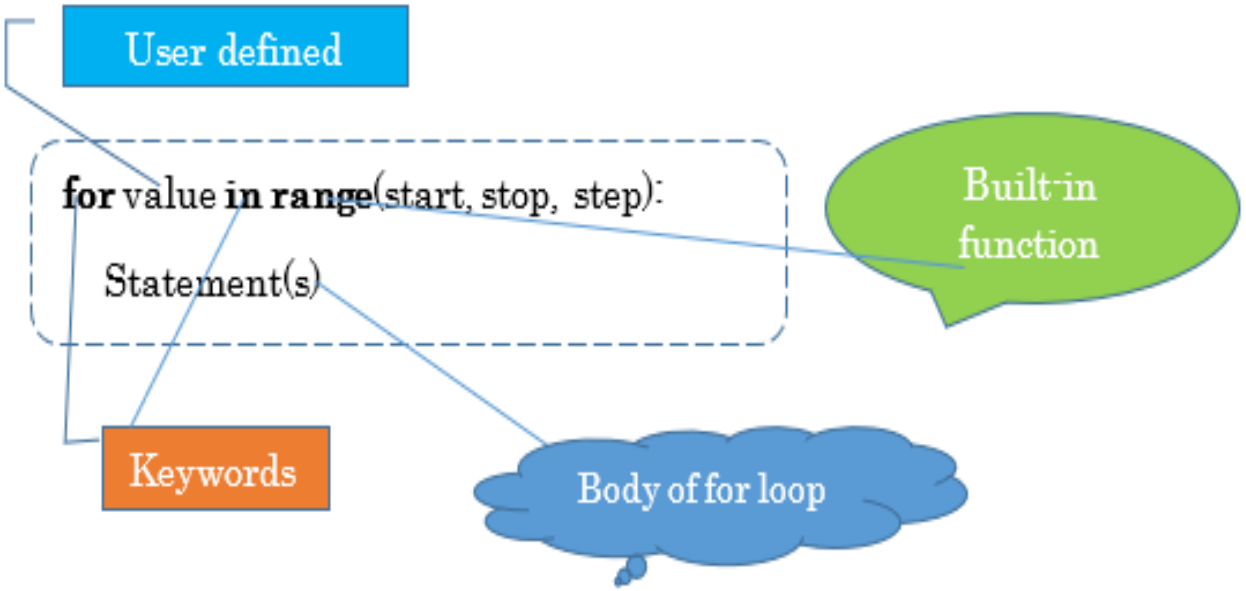
```
for character in "Python":
        print(character)
```

- Output:

```
P
y
t
h
o
n
```

for loop: Type II

- This type of for loop is used to repeat the group of statements for a specified number of times.
- It is associated with range() function.
- For loop syntax with range() :



- **for** is a keyword which begins the loop.
- This loop will be executed for the number of times specified by the built-in function range ().
- **value** is a user defined loop variable that holds each value in the range;
- The statements in the body of the loop will be executed once for each iteration.
- For loop is **count controlled loop** which repeats a block of code for a specific number of times.

range () function

- Python has a built-in function called range (), which generates the integer numbers between the given start integer to the stop integer.
- It is generally used to iterate in the for loop.
- **Syntax:**

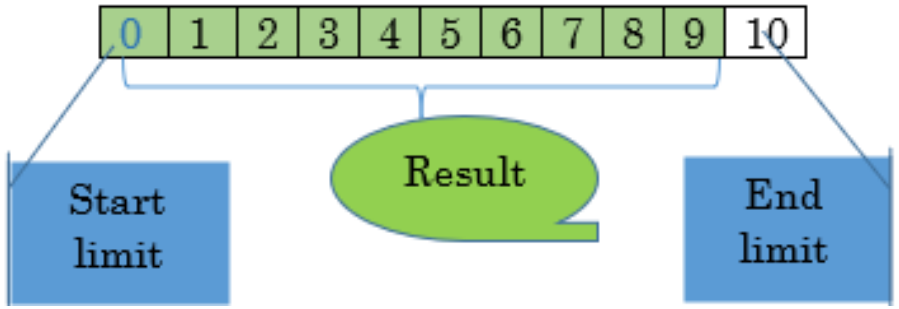
range (start, stop, step)

Function_name

arguments	Description
start	A start argument is a starting number of the sequence. i.e., lower limit. By default, it starts with 0 if not specified.
stop	A stop argument is an upper limit. i.e. generate numbers up to this number, The range () function excludes this number in the result.
step	The step is a difference between each number in the result. The default value of the step is 1 if not specified.

```
>>> range(0,51,5)
range(0, 51, 5)
>>> #To generate numbers from o to 50 with step size 5
>>> list(range(0,51,5))
[0, 5, 10, 15, 20, 25, 30, 35, 40, 45, 50]
```

range(0,10,1)



KUMARAGURU college of technology
character is life

KCED
Learning Augmented

Examples for range()

Example	Output	Description
range(5)	[0, 1, 2, 3, 4]	If only one argument is present, it is considered as stop. Equivalent to (0,5,1).
range(1,5)	[1, 2, 3, 4]	If 2 arguments are present, it is taken as start and stop.
range(0,-10,-1)	[0,-1,-2,-3,-4,-5,-6,-7,-8,-9]	Step is -1 which implies decrement by 1
range(-4,4,2)	[-4, -2, 0, 2]	Lower limit is -4 and upper limit is 4 and difference is 2
range(1,1)	[]	Empty list. Equivalent to (1,1,1)
range(0)	[]	Empty list. Equivalent to (0,0,1)

LEARNING AUGMENTED

Points to remember about Python range() function arguments

- range() function works only with the integers i.e., whole numbers.
- All argument must be integers. You cannot pass a string or float number or any other data type in a start, stop and step argument of a range().
- All three arguments can be positive or negative.
- The step size must not be zero. If a step size is zero Python raises a ValueError exception.
- Start and step are optional arguments. Default value of start is 0 and step is 1
- Result of range function excludes stop limit

KUMARAGURU college of technology
character is life

KCED
Learning Augmented

LEARNING AUGMENTED

Example:Write a program to find the sum of first n natural numbers

```
# To find Sum of N Natural Numbers
N = int(input("Enter any Number: "))    # to read N from user
result = 0  # to store the result
for number in range(1, N + 1,1):
        result = result + number
print("The Sum of Natural Numbers from 1 to", N, "is", result) # to print the result
```

Output:

Enter any Number: 5

The Sum of Natural Numbers from 1 to 5 is 15

KUMARAGURU college of technology
character is life

KCED
Learning Augmented

LEARNING AUGMENTED

Write a program to print the multiplication table(up to 16) of a given number

```
n=int(input("Enter the number for which multiplication table is to be generated:"))
for i in range(1,17):#loop executes from 1 to 16
    print(n,"x",i,"=",n*i)
```

5 x 1 = 5
5 x 2 = 10
5 x 3 = 15
5 x 4 = 20
5 x 5 = 25
5 x 6 = 30
5 x 7 = 35
5 x 8 = 40
5 x 9 = 45
5 x 10 = 50
5 x 11 = 55
5 x 12 = 60
5 x 13 = 65
5 x 14 = 70
5 x 15 = 75
5 x 16 = 80

for loop with else:

- Like while loop, for loop can have an **optional else** block.
- The else part is executed if the items in the sequence used in for loop exhausts.
- In some cases, break statement can be used to stop for loop
- In such case, the else part is ignored.
- Hence, a for loop's else part executes if no break statement occurs.

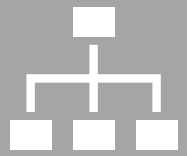
```
#to demonstrate else with for loop
for i in "A loop can have optional else part":
    print(i,end="")
else:
    print("\nThe sentence is read. No character left")
```

Output:

```
A loop can have optional else part
The sentence is read. No character left
```

Nested loops

Loop with in another loop is called nested loop.



The **for** and **while** loop statements can be nested in the same manner as nested if structures.

KUMARAGURU college of technology
character is life

KCED
Learning Augmented

Example

LEARNING AUGMENTED

This example is to print the following pattern which has 5 rows using for loop.

1

1 2

1 2 3

1 2 3 4

1 2 3 4 5

Program

```
# to print the number pattern
n = 6
for row in range(1, n):          # outer loop- loop until it reaches n-1
    for column in range(1, row + 1): # inner loop until it reaches row
        print(column, end=' ')       # print number
    print()                  #new line after each row to display pattern correctly
```

Output

```
1
1 2
1 2 3
1 2 3 4
1 2 3 4 5
```

Explanation:

In the above example, outer loop executes n-1 times.

For each iteration of outer loop, inner loop is executed completely.

Here inner loop is executed for “row” times. i.e. 5 times.

KUMARAGURU college of technology
character is life

KCED
Learning Augmented

Loop control statements

- Loops can be used to execute the block of code repeatedly.
- But sometimes, we may wish to exit a loop or skip a current iteration when it meets a specific condition.
- It can be done using loop control statements.
- Thus, loop control statements can be used to alter or control the flow of loop execution.
- Python supports following loop control statements
  - break
  - continue
  - pass

KUMARAGURU college of technology
character is life

KCED
Learning Augmented

break statement

- The break Statement is a loop control statement which is used to terminate the loop.
- As soon as the break statement is encountered with in a loop, the Python interpreter stops the loop iteration and returns the program control to the statement following loop body immediately.
- If break statement is encountered inside a nested loop, then break will terminate the innermost loop.

LEARNING AUGMENTED

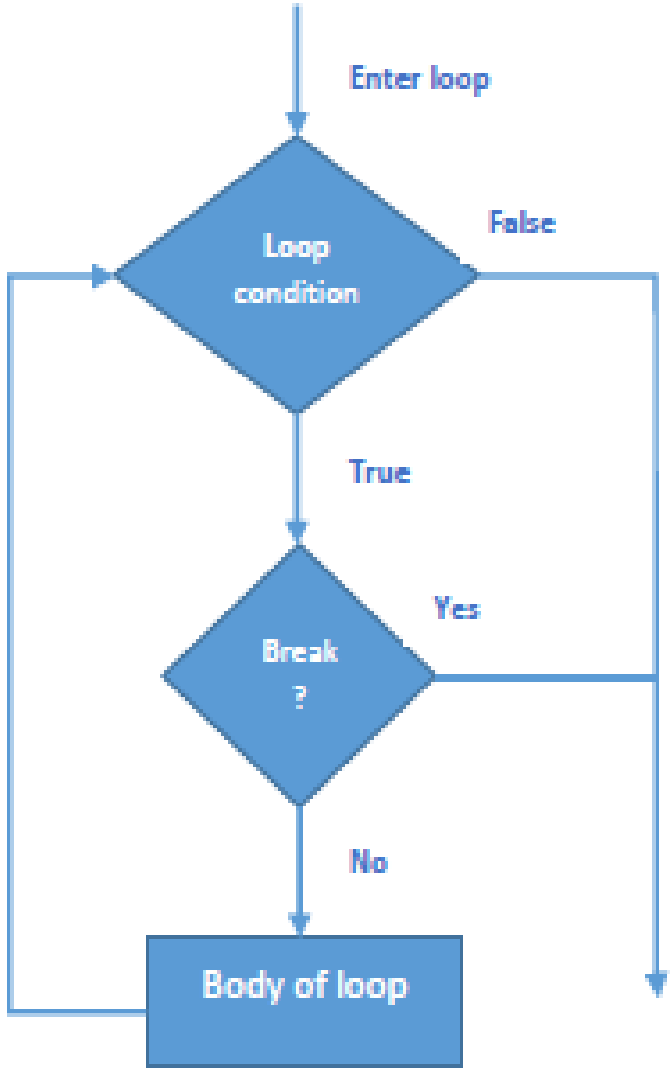
Syntax :

for loop with break statement

```
for var in sequence:
    statement(s) – inside for loop
    if condition:
        statement(s) – if block
        break
    statement(s) – inside for loop
statement(s) – outside for loop
```

while loop with break statement

```
while condition:
    statement(s) – inside while loop
    if condition:
        statement(s) – if block
        break
    statement(s) – inside while loop
statement(s) – outside while loop
```



KUMARAGURU college of technology
character is life

KCED
Learning Augmented

LEARNING AUGMENTED

Write a program to accept the input from user repeatedly until he/she supplies 0 as input

```
#to read a input infinitely until the input is 0
while True:       # infinite loop
    n=int(input("Enter any number : "))
    if n==0:       # condition for break
        break
#statement following body of the loop
print("The loop is terminated as you pressed 0")
```

Output:

```
Enter any number : 5
Enter any number : 6
Enter any number : 9
Enter any number : 4
Enter any number : 0
The loop is terminated as you pressed 0
```

KUMARAGURU college of technology
character is life

KCED
Learning Augmented

LEARNING AUGMENTED

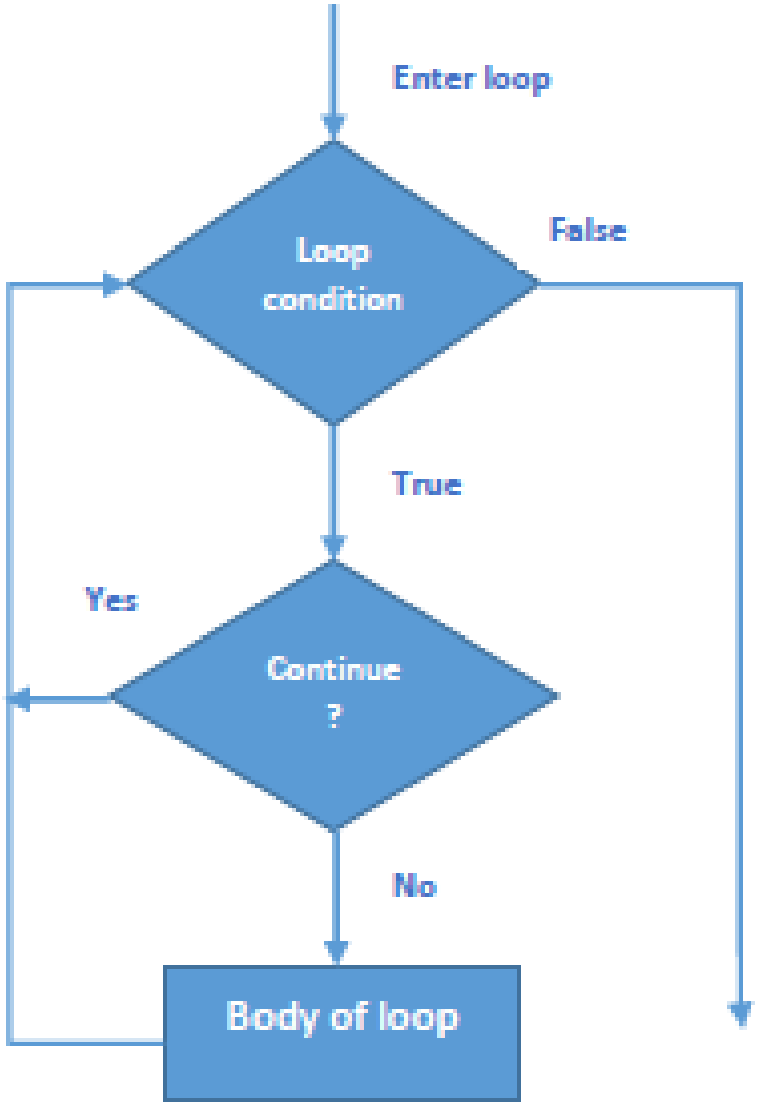
continue statement

- The **continue** statement is a loop control statement, used to skip the rest of the code inside a loop for the current iteration only.
- Whenever the **continue** statement is encountered in a loop, the Python interpreter ignores the rest of statements in the loop body for current iteration and returns the program control to the beginning of loop (condition checking step).
- Thus, it doesn't terminates the loop rather continues with the next iteration.

Syntax

```
for var in sequence:
    statement(s) – inside for loop
    if condition:
        statement(s) – if block
        continue
    statement(s) – inside for loop
statement(s) – outside for loop
```

```
while condition:
    statement(s) – inside while loop
    if condition:
        statement(s) – if block
        continue
    statement(s) – inside while loop
statement(s) – outside while loop
```



KUMARAGURU college of technology
character is life

KCED
Learning Augmented

LEARNING AUGMENTED

Write a program to print all the characters in a string except vowels

```
# To print the characters in a sentence except vowels
string=input("Enter any sentence: ")
print("Actual sentence\n",sentence)
print("sentence without vowels")
for char in sentence:
    if char == 'a' or char=='e' or char=='i' or char=='o' or char=='u:
        continue
    print(char,end="")
```

```
Enter any sentence: you have to dream before your dream can come true
Actual sentence
you have to dream before your dream can come true
Sentence without vowels
y hv t drm bfr yr drm cn cm tr
```

KUMARAGURU college of technology
character is life

KCED
Learning Augmented

LEARNING AUGMENTED

pass statement

The **pass** is a null statement means nothing happens when pass is executed.

It results into no operation.

It is used when a statement is required syntactically but you do not want any code to execute.

Python interpreter ignores a comment entirely, while pass is not ignored as it is a valid statement.

KUMARAGURU college of technology
character is life

KCED
Learning Augmented

LEARNING AUGMENTED

Example

```
# To demonstrate pass statement
# pass is a placeholder for future use
for i in range(1,10):
    pass
```

Output:

(nothing is displayed)

KUMARAGURU college of technology
character is life

KCED
Learning Augmented

LEARNING AUGMENTED

Write a program to generate numbers from 1 to 10 using range() and perform multiplication of the number with 5 if it is an odd number, do not perform any operation if it is an even number.

```
#to demonstrate pass statement
for number in range(1,11): #loop iterates until 10
    if number %2==0:            # condition for even number
        pass                    # no operation
    else:
        print(number * 5,end=" ")  # multiply the number with 5 and prints it
```

Output:

5 15 25 35 45

Example – To illustrate the use of loop with else

```
# program to check if the input number is prime or not
# take input from the user
num = int(input("Enter a number: "))
# prime numbers are greater than 1
if num > 1:
   # check for factors
   for i in range(2,num//2):
       if (num % i) == 0:
           print(num,"is not a prime number")
           break
   else:
       print(num,"is a prime number")
# if input number is less than or equal to 1, it is not prime
else:
   print("Invalid Input")
```

```
Enter a number: 5
5 is a prime number
```

```
Enter a number: 12
12 is not a prime number
```

```
Enter a number: -1
Invalid Input
```

KUMARAGURU college of technology
character is life

KCED
Learning Augmented

LEARNING AUGMENTED

Activity

Write a program to find all numbers between 2000 and 3000 (both inclusive) which are divisible by 7 but not a multiple of 5. All such numbers are to be printed in a comma separated sequence.

Output: 2002, 2009, 2016, ...

Tips

Blue light is basically the rays that are emitted by the digital screen, which may cause severe health problems. Flux software for PCs, Twilight app for Smartphones can act as blue light filters.

Microcosm
The Co-existing Architecture

Kumaraguru Eco Campus

Annually, the **4200 trees** inside the Kumaraguru Campus, on an average, releases **10,92,000 pounds** of oxygen inhaled by **42,000 humans**.

Wellness 360

Find ways to structure and optimize your time for when and where you learn best and keep your **learning on track**

KUMARAGURU college of technology

KCED
Learning Augmented