Need of iterative control

Repeated execution of set of statements

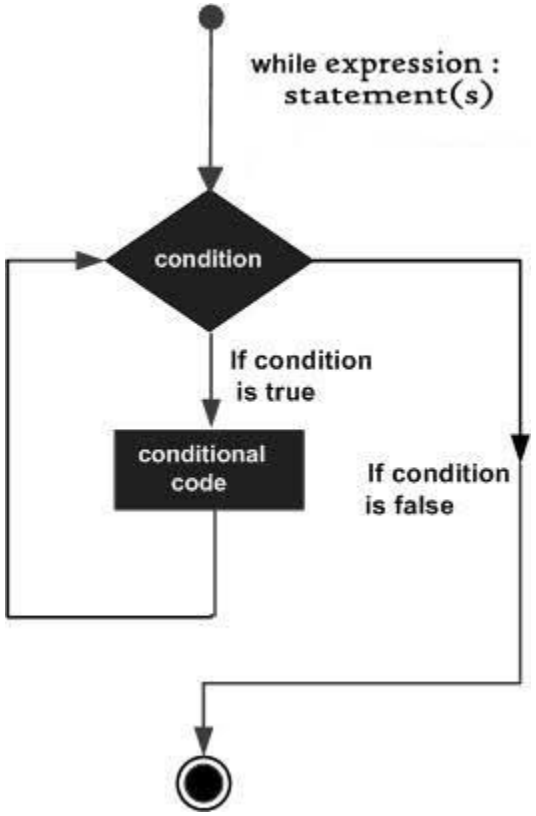
- An **iterative control statement** is a control statement providing repeated execution of a set of instructions
- Because of their repeated execution, iterative control structures are commonly referred to as “loops.”

While statement

- Repeatedly executes a set of statements based on a provided Boolean expression (condition).
- All iterative control needed in a program can be achieved by use of the while statement.

Syntax of While in Python

```
while (test condition):                    # Loop test
        statement 1                        # Loop body
        ……..
        ……..
        statement n
```



Example use

Sum of first 'n' numbers

```
sum =0
current =1
n=3
while  (current  <= n):
    sum=sum + current
    current = current + 1
```

Iteration	sum	current	current <= 3	sum = sum + current	current = current + 1
1	0	1	True	sum = 0 + 1 (1)	current = 1 + 1 (2)
2	1	2	True	sum = 1 + 2 (3)	current = 2 + 1 (3)
3	3	3	True	sum = 3 + 3 (6)	current = 3 + 1 (4)
4	6	4	False	loop termination	

Print values from 1 to 9 in a line

```
i=1
while i < 10:
        print(i)
        i += 1
```

Output:

1 2 3 4 5 6 7 8 9

Include end=‘ ‘ in print statement to suppress default move to new line

Print values from 0 to 9 in a line

```
i=0
while i < 10:
        print(i, end=' ')
        i += 1                    # Or, a = a + 1
```

Output:

0 1 2 3 4 5 6 7 8 9

Include end=‘ ‘ in print statement to suppress default move to new line

Factorial of a number

```
N=int(input())
i=1
fact=1
while i <=N:
        fact=fact*i
         i += 1
print(fact)
```

Print numbers from 1 to n

```
n= int(input(“Enter the number”))
counter=1
while(counter<=n):
        print(counter,end=“ ”)
        counter=counter+1
print(“THE END”)
```

Break, continue, pass, and the Loop else

- break Jumps out of the closest enclosing loop
- continue Jumps to the top of the closest enclosing loop
- pass Does nothing at all: it’s an empty statement placeholder
- Loop else block runs if and only if the loop is exited normally (i.e., without hitting a break)

1. Generating Natural Numbers.
2. Generating Odd sequence.
3. Generating Even sequence.
4. Sum of ‘n’ natural numbers.
5. Sum of ‘n’ i/p numbers.
6. Factorial of ‘n’ numbers.
7. Multiplication table.
8. Fibonacci series (0 1 1 2 3 5 …..)
9. Sum of Digits.
10. Reversing the digits of a number.

Reversing the digits of a number.

```
n=int(input("Enter the number"))
counter=1
total=0
while(counter<=n):
    total=total+counter
    counter=counter+1

print(total)
```

Factorial of ‘n’

```
n=int(input("Enter the number"))
counter=1
factorial=1
while(counter<=n):
    factorial = factorial * counter
    counter=counter+1

print(factorial)
```

#Generate Natural Numbers

```
print("Natural Number")

n=int(input("Enter the number"))

i=0

while(i<n):

    print(i+1)

    i=i+1

print("End")
```

Output:

```
Natural Number
Enter the number4
1
2
3
4
End
```

#Odd Series

```
print(“Odd Series")

n=int(input("Enter the number"))

i=0

j=1

while(i<n):

    print(j)

    i=i+1

    j=j+2

print("End")
```

Output:

```
Odd Series
Enter the number4
1
3
5
7
End
```

#Sum of ‘N’ Natural Numbers

```
print("Sum of n Natural Number")
n=int(input("Enter the number"))
i=1
sum=0
while(i<=n):
    sum=sum+i
    i=i+1
print("Sum=",sum)
```

Output:

```
Sum of n Natural Numbers
Enter the number4
Sum= 10
```

#Sum of 'N' I/P Numbers

```
print("Sum of n I/P Number")
n=int(input("Enter the number"))
i=1
sum=0
while(i<=n):
    a=int(input("Enter the number"))
    sum=sum+a
    i=i+1
print("Sum=",sum)
```

Output:

```
Sum of n I/P Number
Enter the number4
Enter the number10
Enter the number20
Enter the number30
Enter the number40
Sum= 100
```

#Factorial of 'N' Natural Numbers

```
print("Factorial of n Natural Number")
n=int(input("Enter the number"))
i=1
fact=1
while(i<=n):
    fact=fact*i
    i=i+1
print("Factorial of",n,"=",fact)
```

Output:

```
Factorial of n Natural Number
Enter the number5
Factorial of 5 = 120
```

#Multiplication Table

```
print("Multiplication Table")
n=int(input("Enter the table number to be
generated"))
i=1
print("****Table",n,"*****")
while(i<=10):
    print(i, "x", n, "=", i*n)
    i=i+1
```

Output:

```
Multiplication Table
Enter the table number to be generated4
****Table 4 *****
1 x 4 = 4
2 x 4 = 8
3 x 4 = 12
4 x 4 = 16
5 x 4 = 20
6 x 4 = 24
7 x 4 = 28
8 x 4 = 32
9 x 4 = 36
10 x 4 = 40
```

```
#Length of a given number

n=input("Enter the number")

x=len(n)

print("Length of input number is",x)
```

#Length of a given number

```
n=int(input("Enter the number"))

c=0

a=n

while(n!=0):

    n=n//10

    c=c+1

print("Length of", a , "is",c)
```

Output:

Enter the number1234567
Length of 1234567 is 7

#Fibonacci series (0 1 1 2 3 5 …..)

```
n=int(input("Enter the number of items to be generated:"))
a=0
b=1
print(a)
print(b)
i=2
while(i<n):
    c=a+b
    print(c)
    a=b
    b=c
    i+=1
```

Output:

```
Enter the number of items to be generated: 6
0
1
1
2
3
5
```

#Sum of Digits

```
print("Sum of Digits")
n=int(input("Enter the number"))
sum=0
a=n
while(n!=0):
    m=n%10
    sum=sum+m
    n=n//10
print("Sum of Digits of",a,"=",sum)
```

Output:

```
Sum of Digits
Enter the number1234
Sum of Digits of 1234 = 10
```

```
Read a
Assign b=1
x=a&b
if(x==0):
        print(“LSB is not set”)
else:
        print(“LSB is set”)
stop
```

```
n=int(input("Enter the number of elements"))
z=0
p=0
n=0
while(n!=0):
        a=int(input("Enter the number to be tested"))
        if(a>0):
                p=p+1
        elif(a<0):
                n=n+1
        else:
                z=z+1
print("No of Zeros are",z)
print("No of positive numbers are",p)
print("No of Negative numbers are",n)
```

#Reversing the number

```
print("Reversing a number")
n=int(input("Enter the number"))
rev=0
a=n
while(n!=0):
    m=n%10
    rev=(rev*10)+m
    n=n//10
print("Reverse of",a,"=",rev)
```

Output:

```
Reversing a number
Enter the number123
Reverse of 123 = 321
```

#Palindrome

```
print("Palindrome Check")
n=int(input("Enter the number"))
rev=0
a=n
while(n!=0):
    m=n%10
    rev=(rev*10)+m
    n=n//10
if(a==rev):
    print("The number",a,"is a palindrome")
else:
    print("The number",a,"is not a palindrome")
```

Output:

```
Palindrome Check
Enter the number121
The number 121 is a
palindrome
```

```
Palindrome Check
Enter the number1234
The number 1234 is not a
palindrome
```

#Binary to Decimal conversion

```
print("Binary to Decimal conversion")
n=int(input("Enter the Binary number"))
dec=0
k=0
a=n
while(n!=0):
    m=n%10
    dec=dec+(m*(2**k))
    n=n//10
    k=k+1
print("Decimal equivalent of",a,"is =",dec)
```

Output:

```
Binary to Decimal conversion
Enter the Binary number101
Decimal equivalent of 101 is = 5
```

Class Average

- Given marks secured in CSE1001 by the students in a class, design an algorithm and write a Python code to determine the class average. Print only two decimal digits in average

Average marks scored by ‘N’ number of Students

Step 1: Start
Step 2 : Read Number Of Students
Step 3 : Initialize counter as 0
Step 4 : Input mark
Step 5 : Add the mark with total
Step 6 : Increment the counter by 1

Step 7: repeat Step 4 to Step 6 until counter less than number of students

Step 7: Divide the total by number of students and store it in average

Step 8: Display the average

Step 9: Stop

Test Cases

Input

5

90 85 70 50 60

Output

71.00

Processing Involved

Already Know

- To read values from user
- To check if a condition is satisfied
- Print characters

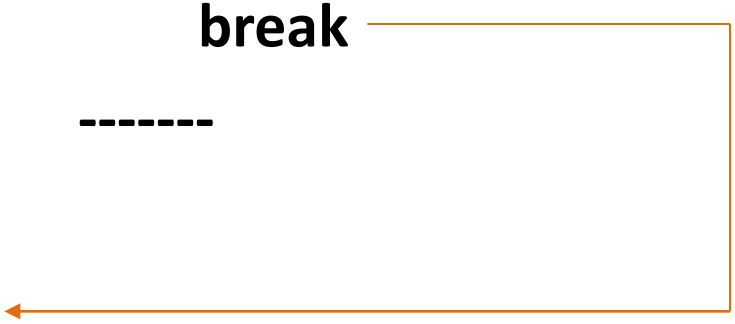
Yet to learn

- Repeatedly execute a set of statements

Break- Loop Terminator

The break statement is used to terminate the loop. When break is executed inside a loop control automatically transferred to the first statement after the loop. Break is usually associated with an if statement.

```
While( condition ):
    -------
    if(condition):
        break
    -------
```



Example

```
i=0

while(i<5):

    if( i == 3):

        break

    print(i)

    i=i+1
```

Continue

The continue statement skips an iteration in a loop. It transfers the control to the beginning of the loop.

```
While( condition ):
    -------
    if(condition):
        continue
    -------
```

Example

```
i=0

while(i<5):

    i=i+1

    if( i == 3):

        continue

    print(i)
```

Break statement

- while True:

```
        name = input('Enter name:')
        if name == 'stop': break
       age  = input('Enter age: ')
        print('Hello', name, '=>', int(age) ** 2)
```

Output:

```
 Enter name:bob
Enter age: 40
Hello bob => 1600
```

Print all even numbers less than 10 and greater than or equal to 0

```
x = 10
while x:
        x = x-1                          # Or, x -= 1
        if x % 2 != 0: continue          # Odd? -- skip print
        print(x, end=' ')
```

```
y = int(input())
if not isinstance(y,int):
    print("Prime number check can be done only for integers")
else:
    if y==0:
        print("Zero is neither prime nor composite")
    elif y<0:
        print("Prime is checked only for positive integer")
    else:
        x = y // 2
        while x > 1:
            if y % x == 0:
                break
            x -= 1
        else:
            print(y, 'is prime')
```

Class Average

```
count =0
total = 0
n=int(input('enter how many mark you want to read: '))
while count < n:
    mark=int(input('enter mark :'))
    if mark<0:
            print ("mark should be greater than 0, terminates.
            break
    total = total + mark
    count = count + 1
else:
    average=total/n
    print("average mark is" , format(average,"0.2f"))
```

Syntax of While in Python

```
while test:                    # Loop test
    statements                 # Loop body
else:                          # Optional else
    statements
# Run if didn't exit loop with break
```

Syntax of While in Python

```
i=0
while(i<5):
    i=i+1
    if( i == 3):
        break
    print(i)
else:
    print("Test")
```

Pattern Generation

- Your teacher has given you the task to draw the structure of a staircase. Being an expert programmer, you decided to make a program for the same. You are given the height of the staircase. Given the height of the staircase, write a program to print a staircase as shown in the example. For example,
  Staircase of height 6:

  #

  ##

  ###

  ####

  #####

  ######

Boundary Conditions: height >0

Pattern Generation

Input	Processing	Output
Staircase height	Create steps one by one To create a step print character equal to length of step	Pattern

Pseudocode

```
READ staircase_height
if staircase_height > 0
x = 1
Repeat
y = 1
Repeat
    print #
    y = y + 1
Until y <= x
x = x + 1
Until x <= staircase_height
End if
Else
Print “Invalid input”
```

Test Cases

Input

3

Output

```
#
# #
# # #
```

Processing Involved

Print step by step

Test Cases

Input

-1

Output

Invalid input

Processing Involved

Boundary condition check fails

Class Average

- Given the marks secured in CSE3041 by the students in a class, design an algorithm and write a Python code to determine the class average. Print only two decimal digits in average.
- Note: If any negative mark is entered, terminate the process with a msg “Invalid input”

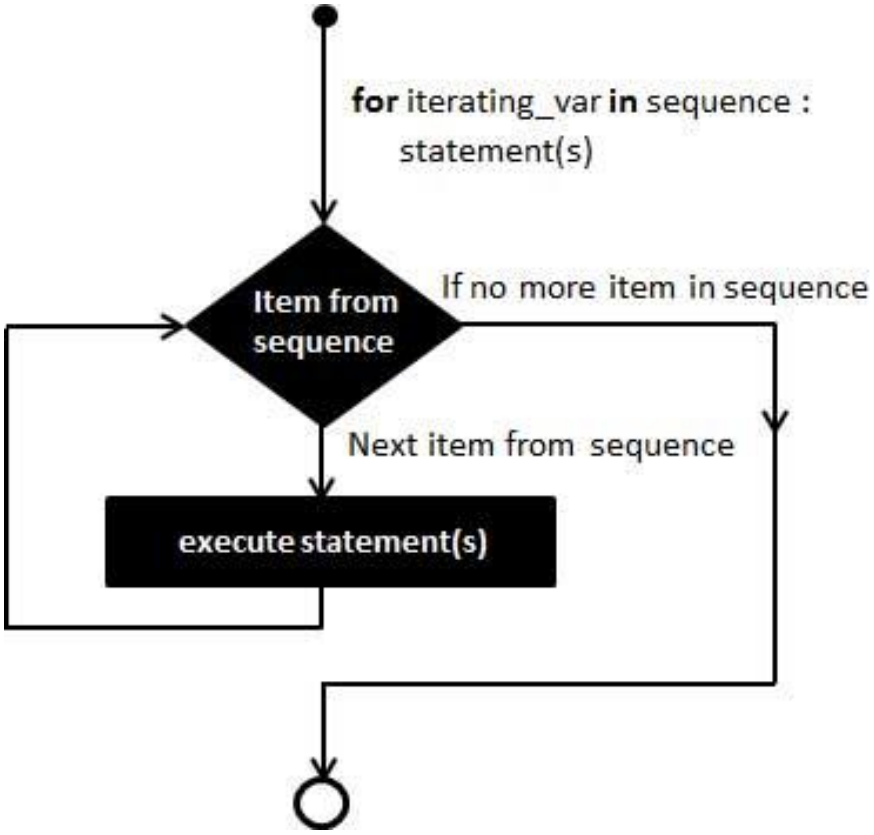
File Edit Format Run Options Window Help

```
count =0
total = 0
n=int(input('enter how many mark you want to read: '))
while count < n:
    mark=int(input('enter mark :'))
    if mark<0:
            print ("mark should be greater than 0, terminates.
            break
    total = total + mark
    count = count + 1
else:
    average=total/n
    print("average mark is" , format(average,"0.2f"))
```

For iteration

- In while loop, we cannot predict how many times the loop will repeat
- The number of iterations depends on the input or until the conditional expression remains true
- While loop is ideal when stop criteria is not explicit

Control flow of for statement



Syntax of for Statement

SYNTAX OF FOR LOOP

```
for target in object:
        statements
        if test:
                break
        if test:
                continue
else:
        statements
```

```
for i in “PYTHON”:
        print(i)

for x in range(10):
        print(x)
```

For and Strings

for iterating_var in sequence or range:

    statement(s)

Example:

```
for i in 'Python':
    print ‘Current Letter :', i
```

For and Strings

When the above code is executed:

```
Current Letter : P
 Current Letter : y
 Current Letter : t
 Current Letter : h
 Current Letter : o
 Current Letter : n
```

For and Range

```
for n in range(1, 6):
    print(n)
```

When the above code is executed:

1

2

3

4

5

range function call

Syntax - range(begin,end,step)

where

Begin - first value in the range; if omitted, then default value is 0

end - one past the last value in the range; end value may not be omitted

Step - amount to increment or decrement; if this parameter is omitted, it defaults to 1 and counts up by ones

begin, end, and step must all be integer values;

floating-point values and other types are not allowed

Example for Range

range(10) → 0,1,2,3,4,5,6,7,8,9

range(1, 10) → 1,2,3,4,5,6,7,8,9

range(1, 10, 2) → 1,3,5,7,9

range(10, 0, -1) → 10,9,8,7,6,5,4,3,2,1

range(10, 0, -2) → 10,8,6,4,2

range(2, 11, 2) → 2,4,6,8,10

range(-5, 5) → −5,−4,−3,−2,−1,0,1,2,3,4

range(1, 2) → 1

range(1, 1) → (empty)

range(1, -1) → (empty)

range(1, -1, -1) → 1,0

range(0) → (empty)

Print Even Numbers Using Range

```
>>> for i in range(2,10,2):
        print(i)
```

Output:

2

4

6

8

Write a program to print all the factors of a given number

```
N = int(input())
for  i in range(1,N+1):
        if(N % i == 0):
                print(i, end=“ “)
```

Write a program to find the sum of all even numbers less than 100.

```
Total=0
for x in range(2,100,2):
        Total+=x
print(Total)
```

1. Check whether the given number is prime or not.

2. Find the GCD of two given numbers

```
#Prime Number
n=int(input("Enter the number"))
prime=1
for i in range(2,n):
    if(n%i==0):
        prime=0
        break

if(prime==1):
    print("Prime NUmber")
else:
    print("Not a Prime NUmber")
```

```
a=int(input("Enter a"))
b=int(input("Enter b"))
prod=a*b
lcm=1
for i in range(a,prod+1):
    if(i%a==0) and(i%b==0):
        lcm=i
        break
print("LCM",lcm)

gcd=1
for i in range(a,1,-1):
    if(a%i ==0) and (b%i==0):
        gcd=i
        break
print("GCD",gcd)
```

GCD of two numbers- Euclid method

```
a=int(input("Enter A"))
b=int(input("Enter B"))
temp=0

while (b>0):
    temp=a%b
    a=b
    b=temp

print(a)
```

Exercise Problem

1. Write a program that read a group ‘g’ of five numbers and another number ‘n’ and print a number in ‘g’ if it is a factor for a given number n?
2. Write a menu driven program which get user choice to perform add/sub/mul/div with the obtained two input?
3. Write a program to display few odd multiples of a odd number n.

Exercise Problem

5. You are given the height of the staircase. Given the height of the staircase, write a program to print a staircase as shown in the example. For example, Staircase of height 6:

```
#
##
###
####
#####
######
```

Boundary Conditions: height >0

6. A store has M kg and N kg of two kinds of apple. The store wants to sell them by filling the two kinds in boxes of equal volumes with no apple left over. Write a program to find the greatest volume of such a box.