Python

Iteration : Loops

- The while statement has the general form:

```
while  condition :
    block
```

- The reserved word while begins the while statement.
- The condition determines whether the body will be executed. A colon (:) must follow the condition.
- block is a block of one or more statements to be executed as long as the condition is true. As a block, all the statements that comprise the block must be indented the same number of spaces from the left

Iteration : Loops

- **Example :** Allow a user to enter a sequence of non-negative integers and sum them. Input should end with a negative no entered

```
a=0
sum=0
Print(“Enter nos to add, -ve no ends the input)
while(a >= 0):
    a=eval(input())
    if a >=0:
        sum=sum+a       #  sum+=a
print(‘Sum is :’ + sum)
```

The break and continue statement

```
Sum=0
while True:
        entry=eval(input())
        if entry < 0:
                break
        else:
                sum+=entry
                continue
Print(“Sum is “, sum)
```

The break and continue statement

```
Sum=0
Done=False
while not Done:
        val=eval(input(“Enter Positive Integers (999 will quit):”))
        if val< 0:
                print(“Negative Value”, val, “ignored”)
                continue  # skip rest of the body for this iteration
        if val !=999:
                sum+=val
        else:
                Done=(val==999)   # 999 input would exit the loop
                continue
Print(“Sum is “, sum)
```

Iteration : The for Statement

- The for statement iterates over a range of values. These values can be a numeric range, or, as we shall, elements of a data structure like a string, list, or tuple.

```
for n in range(1, 11):
    print(n)
```

- The expression range(1, 11) creates an object known as an iterable that allows the for loop to assign to the variable n the values 1, 2, . . . , 10.
- range(begin,end,step)

For loop

- begin is the first value in the range; if omitted, the default value is 0
- end is one past the last value in the range; **the end value may not be omitted**
- change is the amount to increment or decrement; if the change parameter is omitted, it defaults to 1 (counts up by ones)
- begin, end, and step must all be integer values
- floating-point values and other types are not allowed in begin, end or step.

For loop: Example

1.
```
for n in range(21, 0, -3):
    print(n, '', end='')
```
Output: 21 18 15 12 9 6 3

2.
```
for n in range(1000) :
    print(n,end=‘ ‘)
```
Output: 0, 1, 2, . . . , 999.

3.
```
sum = 0
for i in range(1, 100):
    sum += i
print(sum)
```

Output: adds nos from 1 to 99

Iteration : for

- The following examples show how range can be used to produce a variety of sequences:

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)

Nested Loops

Example :

```
1. for i in range(1,5):
        for j in range(1,3):
                print("i=", i," ","j =",  j)
```

This will print as:

```
i= 1 j = 1
i= 1 j = 2
i= 2 j = 1
i= 2 j = 2
i= 3 j = 1
i= 3 j = 2
i= 4 j = 1
i= 4 j = 2
```

Prime Nos using While

```
#to find prime number
n=eval(input("enter number"))
d=2
while d<n/2 :
    if n%d==0:
        print(n," is not a prime number")
        break
    else:
        d+=1
if d>=n/2:
    print(n," is a prime number")
```

Prime No Using For

```
n=eval(input(“enter a number to check”)
for d in range(2,n//2,1):
        if n%d == 0:
                print(n," is Not Prime")
                break
if d == (n//2)-1:
•               print(n, "  is Prime")
```

Nested Loops :Prime Nos within a range

- #print prime nos between a range
- start=int(input("enter the starting no to find as prime"))
- end=int(input("enter the ending no "))

```
for n in range(start,end+1,1):
        for d in range(2,n//2,1):
                if n%d == 0:
                        print(n," is Not Prime")
                        break
        if d == (n//2)-1:
•               print(n, "  is Prime")
```

Nested Loops :Prime Nos within a range

- #print prime nos between a range
- start=int(input("enter the starting no to find as prime"))
- end=int(input("enter the ending no "))

```
for n in range(start,end+1,1):
        flag=1
        for d in range (2,n//2,1):
                if n%d == 0:
                        print(n," is Not Prime")
                        flag=1
                        break
        if flag==0:
•               print(n, "  is Prime")
```

Else in While Loop : Example

```
• start=int(input("enter the starting no to find as prime"))
• end=int(input("enter the ending no "))
• for n in range(start,end+1,1):
•     for d in range(2,n//2,1):
•         if n%d == 0:
•             print(n," is Not Prime")
•             break
•     else:
•         print(n, " is Prime")
•         #if d == (n//2)-1:
•            #print(n, "  is Prime")
```

Exercise

- Request a number from the user. It should then print a multiplication table of the size entered by the user; for example, if the users enters 15, a 15 × 15 table should be printed. Print nothing if the user enters a value lager than 18. Be sure everything lines up correctly, and the table looks attractive.