#29-MAY-22

7th Class - "Python"

WHILE LOOP - (2nd Part) & FUNCTION BASIC

In [4]:
```
                                        # Function

# lets suppos, I have to perform some short of a operation like addition of 2 numbers.

#      To do that, If I have to create a function. How w ecan create that simple possible function.
```

In [ ]:
```
#  Lets Understand :-

# Incase of Python any point of time, whenever we create our own function there is a keyword called as def.

# Definition of function, we can use function as a initiation of function as a def by which we will able to
# define, our own function.

# IF I used reserved keyword as a def & then I can write the function name - it can my name & others later
# we will learn about there is naming Convention we used to follow while creating a fundtion.
```

In [5]:
```
# As of now we can try to give any name while practicing, very first keyword that we all suppose to use is that
# we can define our own function name.

# def test():

# lets suppose I have given after def I have given Test as a fucntion name & then open bracket & Closed bracket
# then we suppose to give a column.

# These is the minimum requiredment to create any kind of a fucntion in Python.
# If we talked about this thing in JAVA or C++.

# Yes, everywhere we try to create our own functions , but procedure to create a fucntion
# in different
# in different language are different, its not same as in Python. As compare to other language
# in Python it very easy to create.
```

In [6]:
```
# So, start with the def give a fucntion name open bracket () & Close bracket & then give a column :

# Then we can write our oen syntax or oueb own logic, whatever implemenation that, I would like to take,
# we can write it down over here.

# def test ()
#     print("this is my first fucntion")

# Once we execute it, we can do that, but how we will be utilize this fucntion, because as of now,
# we have written this function but I am able to get any kind of a outcome.

# we get outcome only at if we are going to call this function. Unless & Untill, if we are not going call
# this fucntion Test(): we will not going to get an outcome.
# Now, if we execute this fucntion.
```

In [1]:
```python
# Code No - 1 :

def test():
    print("this is my first function")
```

In [2]:
```python
test()
```
```
this is my first function
```

In [3]:
```python
# Code NO - 2 :

# Print always return a "nonType" object at any point of a time.

# It is "Nottype" object & we are trying to perform Append operation.

# Why we are geeting an Error because :-

# As we know that, if we are trying to perform an Append operation with the string(str). It is suppose to be
# string(str)

# otherwise, it is going to give us an Error. That is the region as a outcome we are getting an error.


def test():
    print("this is my first fucntion")

test() + "vijay"
```
```
this is my first fucntion
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[3], line 17
     14 def test():
     15     print("this is my first fucntion")
---> 17 test() + "vijay"

TypeError: unsupported operand type(s) for +: 'NoneType' and 'str'
```

In [ ]:
```
# Code No 3 :-

def test1 ():
    return "this is my first function"
```

In [ ]:
```
test1()
```

In [4]:
```
type(test1())
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[4], line 1
----> 1 type(test1())

NameError: name 'test1' is not defined
```

In [5]:
```
test1() + "vijay"
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[5], line 1
----> 1 test1() + "vijay"

NameError: name 'test1' is not defined
```

In [6]:
```
# Note :-

# Difference between "Return Type" & Return Type ( Print)

# 1.Return Type :- Return Type will return as it is, the data type which we have.

# 2.Return Type(print) :-  Print always return an "Non Type"

# Note :-

# Later we will not going to use that much "print statement", we wil going to use a "Logger".
# Logger is standard approach that we follow across the industries not a "Print" at all.


# Note :-

# Return will return as it is, the object that we have, other hand, print will return always a "None Type".

# So, always whenever we are trying to use a return statement, instead of using print statement
# if we looking for any kind of a outcome of our function.
```

In [7]:
```
# Code No 4 :-

def test2():
```

```
  Cell In[7], line 3
    def test2():
                ^
SyntaxError: incomplete input
```

In [8]:
```
# We use "Pass" then if I don't want to mention anything in that case, I Would like to use a "pass".

def test2():
    pass
```

In [9]:
```
# Code No 5:-

def test3():
    return 1,3,5,[1,2,3,4,5]
```

In [10]:
```
test3()
```

Out[10]:
```
(1, 3, 5, [1, 2, 3, 4, 5])
```

In [11]:
```
# Note :-

# If we are trying to return only one thing that is also we can do.

# Return "this is my first function"

# or
# If we are trying to return a multiple things also we can do it.
# Return 1,3,5,[1,2,3,4,54,5]
```

In [12]:
```
# Code No 6 :- Here Lets suppose I would like to hold entire dataset in some of the variable "a"
#                 Yes I am able to hold this entire "tuple" inside a variable ["a"].

a = test3()
a
```

Out[12]:
```
(1, 3, 5, [1, 2, 3, 4, 5])
```

In [13]:
```
# Explanation for below code :-

# We can try to define a variable values to a  multiple variable at a time or we can do a single variable
# at a time.Both approach are feasible ( possible to easily or Conveniently)

# We will be capture a return of a any function in a single variable or in a multiple variable.
```

In [14]:
```
# Code No 7 :-

# Here there might be situation where I would like to hold these entire return whatever my function is
# giving me,I would like to hold this entire return into a multiple variable this way.

# a,b,c = 23,45,56 or a,b,c = (23,45,56) - I can try to enclose this into a Tuples.



a = 10
b = 20
c = 30
```

In [15]:
```
a,b,c = (23,45,56)
```

In [16]:
```
a,b,c,d = test3()
```

In [17]:
```
a
```

Out[17]:
```
1
```

In [18]:
```
b
```

Out[18]:
```
3
```

In [19]:
```
c
```

Out[19]:
```
5
```

In [20]:
```
d
```

Out[20]:
```
[1, 2, 3, 4, 5]
```

In [21]:
```
# Code No 8 :-

# Let suppose, I would like to write a function which will do a multiplication as well as additional operation.

def test4():
    a = 4*5
    b = 6+4
    return a,b
```

In [22]:
```
test4()
```

Out[22]:
```
(20, 10)
```

In [23]:
```
g = test4()
```

In [24]:
```
g
```

Out[24]:
```
(20, 10)
```

In [25]:
```
j,k = test4()
```

In [26]:
```
j
```

Out[26]:
```
20
```

In [27]:
```
k
```

Out[27]:
```
10
```

In [28]:
```
# Code No 9:-

# I would like to write function like to write a function which will do a multiplication as well as additional

def test4() :
    a = 4*5
    b = 6+4
    return a,b
```

In [29]:
```
test4()
```
Out[29]:
```
(20, 10)
```

In [30]:
```
g = test4()
```

In [31]:
```
g
```
Out[31]:
```
(20, 10)
```

In [32]:
```
j,k = test4()
```

In [33]:
```
j
```
Out[33]:
```
20
```

In [34]:
```
k
```
Out[34]:
```
10
```

In [35]:
```
# Code No 10

# Note :- SO, Underscore (_) is called as place holder. I am not defining any kind of a real name or any kind
# of named Variable over here, instead of that, I have just used underscore (_,_,_), which is a place holder.

#In "Function" we can rewrite the existing code, again & again.


_, m = test4()
```

In [36]:
```
m
```
Out[36]:
```
10
```

In [37]:
```
_
```
Out[37]:
```
20
```

In [38]:
```
test3()
```
Out[38]:
```
(1, 3, 5, [1, 2, 3, 4, 5])
```

In [39]:
```
_,_,_,g = test3()
```

In [40]:
```
g
```
Out[40]:
```
[1, 2, 3, 4, 5]
```

In [41]:
```
# Code No 10 :-

# Here we are able to understand that, this is the while loop and it will start from (1 fo till 9) or

# (1,3,5,7,9) Then it will able to perform multiple operation.

#If  I have to convert this below entire code into a function, because the uses of Function is to reuseability.

a = 1
b = 10
while a<= b:
    print(a)
    a = a+2
else :
    print("print this else block")
```
```
1
3
5
7
9
print this else block
```

In [42]:
```
# Code No 11 :-
# Here upper we have the "While Loop" convert into"Function".

# Here why outcome is an Error because, it is trying to tell me that, I am not able to find our "a"
# but you are using "a" in a function.

# Syntaxwise there is no issue with the function. We are suppose to pass this data, while I was working with
# "while Loop" I have pass the below "value" while working with while Loop.

# a = 1
# b = 10
# Here inside "function" I have not pass any kind of value. So, that is the region it is giving me an error.


def test5():
    while a<=b :
        print(a)
        a = a+2
    else :
        print("print this else block")
```

In [43]:
```
test5()
```

```
---------------------------------------------------------------------------
UnboundLocalError                         Traceback (most recent call last)
Cell In[43], line 1
----> 1 test5()

Cell In[42], line 18, in test5()
     17 def test5():
---> 18     while a<=b :
     19         print(a)
     20         a = a+2

UnboundLocalError: cannot access local variable 'a' where it is not associated with a value
```

In [44]:
```
# Code No 12 :-


# By putting this below "Value"

# a = 1
# b = 10

# By putting this If I am call "test5()" as outcome we can see exactly same outcome over here in function also.



def test5():
    a = 1
    b = 10
    while a<= b :
        print(a)
        a = a+2
    else :
        print("print this else block")
```

In [45]:
```
test5()
```

```
1
3
5
7
9
print this else block
```

In [46]:
```
# Code No 13 :-

# Here in this code lets suppose I would like to pass this below "value"

# a = 1
# b = 10

# a & b dynamically, I do not wants to keep this value inside my function.

# Last code we have hard coded the value of "a & b " at time function call.

# If I want to keep 10,20,100 or 1000 argument I can keep over here, but here at function call
# I can try to pass

# value of "a,b" & then it is going to work. What change I have done over here is I have remove the "value"

# a = 1
# b = 10

# Then i tried to pass these test5(a,b) as an argument.

# Here 1 line was giving me an "Error" why because signature of function test5(a,b) : is says that
# whoever is going to
# pass the value "a & b". certain value they are suppose to pass.That was the region its giving me an "Error"


def test5(a,b):
    while a<=b :
        print(a)
        a = a+2
    else :
        print("this is else block")
```

In [47]:
```
test5()
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[47], line 1
----> 1 test5()

TypeError: test5() missing 2 required positional arguments: 'a' and 'b'
```

In [48]:
```
test5(1,10)
```

```
1
3
5
7
9
this is else block
```

In [49]:
```
# Code No 14 :-

# May be I would  like to call these function test5(3,56) may be with different value.

# The value of a = 3
# The value of b = 56

# As a outcome we can see I can pass any value over here, entire piss of code will going to work accordinly.
# So, whatever value of "a & b" we have written it is going to perform, at the same operation.
# Here in last line code - Here it is giving me any kind of outcome with the help of "print statement"

#"Print" always "return" "NonType" basically whatever outcome it is trying to give it to me, outcome are
# Basically "Nontype".


test5(3,56)
```

```
3
5
7
9
11
13
15
17
19
21
23
25
27
29
31
33
35
37
39
41
43
45
47
49
51
53
55
this is else block
```

In [50]:
```
test5(30,2)
```

```
this is else block
```

In [51]:
```
type(test5(30,2))
```

```
this is else block
```

Out[51]:
```
NoneType
```

In [52]:
```
# Note :-

# Definition of Function :-
# Function is nothing but it is just a reaper top of the logic, which is going to be increase the reuseability
# code at any point of a time.
```

In [53]:
```
# Code No 15 :-

#

def test5(a,b):
    while a<= b:
        return a
        a = a+2
    else :
        return "print this else block"
```

In [54]:
```
test5(1,10)
```

Out[54]:
```
1
```

In [55]:
```
type(test5(30,2))
```

Out[55]:
```
str
```

In [56]:
```
# Code No 16 :-

#Explaination of below code:-

#Here this line code - ( Here I have deactivated "return" & activated "Print" Statement).

#Here incase of "Print" I have written Print (a) or return (a) & a = a+2.

# Now, things has changed, if i will comment this out (#return a ) If I am going to execute it.
# Can we say that I am able to get outcome. - YES -  (1 3 5 7 9 'print this else block').
```

In [57]:
```
def test5(a,b):
    while a<=b :
        print(a)
        #return a
        a = a+2
    else :
        return("print this else block")
```

In [58]:
```
test5(1,10)
```

```
1
3
5
7
9
```

Out[58]:
```
'print this else block'
```

In [59]:
```
# Code No - 17 :-

#Explaination of below code:-

#Here this line code - ( Here I have deactivated "Print"(a) & Activate "return" ).

#If we # hastage "print or return" It will deactive automatically.

#Incase of "return" statement as a outcome it is only giving me an "1"
#Why we are geeting only "1" as a outcome. In this code it it trying to "return" something #return a.

#So in a very firts place the value of (a) is (1) or value of (a = 1).

#We are trying to "pass" the value of (a=1).

# It is trying to "Return a" & then it will stop.

# Incase of "return" once it will be get a return, it will not even look for the next one.

# Because, it will "return" & close, because we are able to get atleast one return that is "1"
```

In [60]:
```
def test5(a,b):
    while a<=b :
        #print(a)
        return a
        a = a+2
    else :
        return("print this else block")
```

In [61]:
```
test5(1,10)
```

Out[61]:
```
1
```

In [62]:
```
# Code No - 18 :-
# Lets suppose I am lookimg for all the data 1 upto 10 then.

# For that I have to create a "List" L = [] , may be I can create empty list.

#Here each & everytime I can try to call  l.append(a).

# Here we just trying to return one single entity at a time. So, we can return any multiple thing
# we can try to return over here.

# But once it encounter, the return it will just return out of the function & then it will close.
```

In [65]:
```
def test5(a,b):
    l = []
    while a<=b :
        #print(a)
        l.append(a)
        a = a+2
    else :
        print("print this else block")
    return l
```

In [66]:
```
test5(1,10)
```

```
print this else block
```

Out[66]:
```
[1, 3, 5, 7, 9]
```

In [68]:
```
# Code No - 19 :-

# lets understand Below code:

# Lets suppose there is "List", I have over here inside a "List", i have dataset.

#1. [4,5,6,6,7,7,8,8,9,9,9] This is a integer 2.[3,4,5,6,7,7] - This is "List" & 3."vijay" - This is a "String"

# Requiredment :- Sir is expecting from us is that, write the "Function" which can filter out all the "Integer"
# from any list.Just "integer" part, not a "List" inside "List" not the string.

# what we already have inside this code is that - ("integer, "list" & "String").

# Sir is asking is that write a Logic, where we again & again write a code line by line everyday so the same.

# But, he is looking for different approach. He is looking for "Function Approach", So, that again & again
# we don't have to write, whenever it is required, just call that function & then my work is done.

# If I am looking for such kind of approach ,than below is the code that I have written.

# Below I will be able to get "L1" which is just holding an "integer" - [4,5,6,6,7,7,8,8,9,9,9]

# But incase if i am looking for generic approach,I am looking for functional approach.

# Convert this entire this things in a function. So, where we can pass any "List" & for any kind of "List"
# it will be able to perform same things.
# So, here what I did is that, I have create a "Function" with the help of "def" & I can write  "Test7():"
```

In [ ]:
```
# test7() - Code Explaination :-

# So, here If i am trying to execute this code, everythings looks well & Good but the problem is .I am not able
# outcome.

# If I have to get any kind of =outcome on a Functional call, what I am suppose to do. I have 2 options.

# 1."Print" & 2nd Option is - "return"

# 1."Print" :- Print always try to print a "Non Type".

# 2. "Return" :- Return will try to return the actual data type.
```

In [71]:
```
l = [4,5,6,6,7,7,8,8,9,9,9,[3,4,5,6,7,7],"vijay"]
l1 = []
for i in l :
    if type(i) == int :
        l1.append(i)
```

In [74]:
```
l1
```

Out[74]:
```
[4, 5, 6, 6, 7, 7, 8, 8, 9, 9, 9]
```

In [83]:
```
def test7() :
    l1 = []
    for i in l :
        if type(i)==int :
            l1.append(i)
```

In [84]:
```
l1
```

Out[84]:
```
[4, 5, 6, 6, 7, 7, 8, 8, 9, 9, 9]
```

In [85]:
```
test7()
```

In [ ]:
```

```

In [ ]:
```
# Code No - 20 :-

# Lets suppose on this code, if we add "returnL1" then

# So, now here if we call "test7" as outcome we can see - [4, 5, 6, 6, 7, 7, 8, 8, 9, 9, 9]

# Here I am trying to return, so I am trying to take something from the "Function" that is why,
# we are using "Return"

# Rest the logic are as it is. we did not modify or change any kind of logic al all.
```

In [86]:
```
def test7() :
    l1 = []
    for i in l :
        if type(i)==int :
            l1.append(i)
    return l1
```

In [87]:
```
test7()
```

Out[87]:
```
[4, 5, 6, 6, 7, 7, 8, 8, 9, 9, 9]
```

In [88]:
```
# Code No - 21 :-

# Explanation :-

# lets suppose I would like to utilized this "fucntion" for any kind of 'List'.For any kind of 'List"
# I would like to utilize same "function". So here I have a "List" - L3.

# We have to make changes inside the function, again & again.I have to go & Change a "Function" over there.

# At same time, I have to keep on changing the variable name & all those things."No" that is not the approach,
# we are suppose to flow, let's make it genric.

# What we can do is we can paas the "L". -  def test7(l)

# We can pass "L" as a argument over here & "L" can be anything.
# whatever data we are going to pass it is going to work on the same way.we have made this things as a generic.

# I can call the test7 and if IU would liek to to test with "L3".If I will pass "L3" according to "L3"
# it it giving me an answer.

# So, we have just made a generic by passing an arguement.So, that user will be able to pass there own "List"
# at any point of a time & they will be perform that Operation.
```

In [95]:
```
l3 = [345,45,5,5,6,5,"vijay",(4,6,6)]
```

In [100…
```
def test7(l):
    l1 = []
    for i in l :
        if type(i) == int :
            l1.append(i)
    return l1
```

In [101…
```
test7(l3)
```

Out[101]:
```
[345, 45, 5, 5, 6, 5]
```

In [102…
```
# Code No - 22:-

# Here I am working with this particular 'Tuples'. Now I would like to create a generic "Function".

# So,where anyone can pass a " Tuples" any kind of "Tuples" & it is suppose to return the same things or
# same outcome.

# If I am looking for below kind of a outcome.

# index of 7 is -2
# index of 6 is -4

# That particular outcome is simple, I have converted this entire code into a "Function".That is my objective.
```

In [2]:
```
t = (3,4,5,6,67,7,87)
a = -1
while a>=-len(t):
    if t[a]== 6 or t[a] == 7:
        print('index of',t[a], "is", a)
    a = a-1
```

```
index of 7 is -2
index of 6 is -4
```

In [8]:
```
# Code No - 23:-

# Def or Function (test8) I can write, i wanted to make it generic by passing an argument (t) into it.

# same piece of code, I have taken over here.

# As a outcome is same, if we compare both above & below this code my outcome is same.

# Without "Def fucntion" my outcome is same & With Fucntion ( Def ) my outcome is same outcome end of the day.
```

In [ ]:
```
# Explanation :-

# But advanatage wise with function (Def). Going forward instead of passing this (t) test(t).

# or we can call this as a  test8(t) or test8(3,4,5,6,67,7,87)
```

In [9]:
```
def test8(t) :
    a = -1
    while a>=-len(t):
        if t[a]== 6 or t[a] == 7:
            print('index of',t[a], "is", a)
        a = a-1
```

In [10]:
```
test8(t)
```

```
index of 7 is -2
index of 6 is -4
```

In [12]:
```
# Definitsation :-

# Whatever code that we have written, so far so forth, we try to convert each & every piece of code
# as a function.

# From normal code we usually write and any code we write, we can convert that code into a function "Def"
```

In [ ]:
```
#   Class task check once in question has provided
```

In [ ]:
```

```

In [4]:
```
                            # It a task to solve during a class Question :-

#1.Try to print this by using while Loop.

#*
#**
#***
#****
#*****
#******
#*******
#********
#*********


#2. Try to Print below by using while Loop :-

# A
# BH
# CIN
# DJOS
# EKPTW
# FLQUXZ
# GMRVY

#3. Try to print all the number divisible by 3 in between a range of 40-400.

#4. Try to filter out all the vowels form below text by using while Loop :

# """Python is a high-level, interpreted, general-purpose programming language. Its design Philosophy
# emhasizes code readability.
# -Python is dynamically-typed and garbage - collected. It supports multiple programming paradigms, including
# structured (particular.....)

# -Guido van Rossum began working on Python in the late 1980's as a successor to the ABC programming language
# and first released ....

# - Python consistently ranks as one of the most popular programming language"""

#5. Try to generate all the even number between 1-1000.

#6. Define a function for all the above problem statement.
```

```
#7. Write a code to get a time of your system.

#8. Write a code to fetch date form your system.

#9. Write a code to send a mail to your friend.

#10. write a code to trigger alarm for your at scheduled time.

#11. Write a code to check IP adress of your system.

#12. Write a code to check a perticular installation in your system.

#13. Write a code to convert any text in to voice.

#14. You have to write a fun which will take string and return a len of it, without using a inbuild fun Len.

#15. Write a fun which will be able to print an index of all premitive element which you will pass.

#16. Write a fun which will take input as a dict and give me out as a list of all the values even in case of
# 2 level nesting it should work.

#17. Write a function which will take multiple list as a input and give me concatnation of all the element
# as and output.

#18. Write a function which will would return list of all the file name from a directory.

#19. Write a function which will be to able to read a image file & show it t you.

#20. Write a function by which you will be able to append two PDF files.

#21. Write a function which can help you to filter only word files from a directory.

#22. Write a function which can read video file & play for you.

#23. Write a function which will be able to shutdown your system.

#24. Write a function which will would return list of all the file name from a directory.
```

In [ ]:

In [ ]:

Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js