In [190]:
```
# THEORY QUESTIONS
```

In [ ]:
```
##  1. What is the difference between a function and a method in Python?
```

In [13]:
```
# ANS:>

# A function is a set of instructions which are independent in nature
# A function is called with its name only to which it is defined.
# A function is called to return the value or data

# A method is a function which is linked to any object.
# A method is called with the given object.
# a method is used to call or handle the data of the class or it have the class.

# Ex.=>

def addition(a,b):
    print("after addition of number is:")
    return a+b
```

In [14]:
```
addition(5,33)
```

```
after addition of number is:
```
Out[14]:
```
38
```

In [31]:
```
# Ex for method

class robin():
    def my_method(self):
        print("method ")
```

In [32]:
```
name = robin()
name.my_method()
```

```
method
```

In [33]:
```
#  2. Explain the concept of function arguments and parameters in Python.
```

In [38]:
```
# ANS:>

# Arguments are passed in the function.arguments are given while we defining the functio
# Parameters are called while we calling the function.These give the values while we wan

# Ex:>

def add_nums(a,b):## here it is argument
    return a+b
```

In [39]:

```
add_nums(6,9)
```

Out[39]:
15

In [41]:
```
add_nums(a=5,b=7)# Here it is parameter as we calling the function and fix the values of
```

Out[41]:
12

In [42]:
```
#  3. What are the different ways to define and call a function in Python?
```

In [1]:
```
# ANS:>
#1
def robin():
    print("by defalut")
```

In [2]:
```
robin()
```

by defalut

In [7]:
```
# 2 Using arrguments:
def Abc(a,b):
    print ("subtraction of nums")
    return a-b
```

In [8]:
```
Abc(9,4)
```

subtraction of nums
Out[8]:
5

In [38]:
```
# 3 Return statement:-

def square(n):
    return n*n
    return(remove.it)
    print(n*n)
    square(5)
    squared_value = square(5)
    squared_value
```

In [39]:
```
square(5)
```

Out[39]:
25

In [41]:
```
#4 lamda:-
```

```
multiple_8 = lambda n:n*8
multiple_8(6)
```

Out[41]:
```
48
```

In [42]:
```
# 4 What is the purpose of the `return` statement in a Python function?
```

In [ ]:
```
# ANS
#return statement is used to give us the value of squares,ut helps us to find multiple o
# EX:
```

In [43]:
```
def square(n):
    return n*n
    return(remove.it)
    print(n*n)
    square(5)
    squared_value = square(5)
    squared_value
```

In [46]:
```
square(8)
```

Out[46]:
```
64
```

In [47]:
```
#  5. What are iterators in Python and how do they differ from iterables?
```

In [48]:
```
# ANS :> Iterables is a kind of objecct that is already stored in python and it give the
# iterators are used to iterate over an iterable object using the next(() function, It a
# EX:>

nums = [1,2,3,4,5]
for i in nums:
    print(i)
```

```
1
2
3
4
5
```

In [51]:
```
# EX:>

a = nums
a = iter(a)
print(a)
```

```
<list_iterator object at 0x0000027B34EA6C80>
```

In [52]:
```
print(next(a))
```

```
1
```

In [53]:

```
a = nums
a = iter(a)
print(next(a))
```

```
1
```

In [54]:

```
print((a))
```

```
<list_iterator object at 0x0000027B34EA6C80>
```

In [55]:

```
print(next(a))
```

```
2
```

In [56]:

```
#  6. Explain the concept of generators in Python and how they are defined.
```

In [65]:

```
# ANS:> generators are used as a code to access the large files to take thee result in a

# EX:>
```

In [72]:

```
def fib(limit):
    a, b = 0, 1
    while b < limit:
        yield b
        a, b = b, a + b
```

In [73]:

```
fib(20)
```

Out[73]:

```
<generator object fib at 0x0000027B35631CB0>
```

In [81]:

```
def fib(limit):
    a, b = 0, 1
    while b < limit:
        yield b
        a, b = b, a + b
```

In [83]:

```
gen = fib(20)
```

In [84]:

```
next(gen)
```

Out[84]:

```
1
```

In [85]:

```
next(gen)
```

Out[85]:

```
1
```

In [86]:

```
next(gen)
```

Out[86]:
```
2
```

In [87]:
```
next(gen)
```

Out[87]:
```
3
```

In [88]:
```
#  7. What are the advantages of using generators over regular functions?
```

ANS genersters are mainly used to wokr with the large files, Generators are used in code to get only limited types of object from any condition which we have passed in the code # generators are have some special commands which are used for regular functions because of its nature of memory cration in system. It not store the value during the conditions we are creating the function but stores the value while we call the function or generator.

In [122]:
```
# ex
gen_local = (i*5
    for i in range(10)
        if i%2 == 0)
```

In [123]:
```
for i in gen_local:
    print (i)
```

```
0
10
20
30
40
```

In [124]:
```
# 8. What is a lambda function in Python and when is it typically used.
```

ANS :> A lambda is a small anonymos function,it is defined without a name which is take any number of arguments and only return one expression.

In [129]:
```
# Ex:>
square_num = lambda n: n*n
```

In [131]:
```
square_num(8)
```

Out[131]:
```
64
```

In [132]:
```
# 9. Explain the purpose and usage of the `map()` function in Python.
```

ANS:> In puthon it is used to work as an iterator to return a result after applying on any function to every item of an iterables like tuples,list.

In [133]:
```
# EX:>
text = ["abc", "bca", "efg"]
map(len,text)
```

Out[133]:

```
<map at 0x27b35638220>
```

In [134]:

```
text
```

Out[134]:

```
['abc', 'bca', 'efg']
```

In [136]:

```
list(map(len,text))
```

Out[136]:

```
[3, 3, 3]
```

In [138]:

```
def text(a,b,c):
    return a+b+c
```

In [144]:

```
x = map(text, ('a'),('b'),('e'))
print(x)
print(list(x))
```

```
<map object at 0x0000027B34EA6230>
['abe']
```

In [145]:

```
#  10. What is the difference between `map()`, `reduce()`, and `filter()` functions in P
```

In [166]:

```
# ANS :> all three functions allows to iterate over each item in an iterable.
# map function can be used to perform mathematical operations of two or more lists or ar
# EX=
def mul_nums(num):
    return num*3
```

In [167]:

```
num = map(mul_nums, [1, 8, 5, 7, 54, 66, 42])
```

In [168]:

```
    print("multiplying of 3 in list")
    for i in num:
        print(i)
```

```
multiplying of 3 in list
3
24
15
21
162
198
126
```

In [174]:

```
# Reduce function iteratos through each items ina list or other iterable data types and

# EX:>

from functools import reduce
def addnums(a, b):
```

```
    return a+b
```

In [175]:

```
alist = [24, 4, 3, 6, 45, 34]
```

In [176]:

```
print("sum of all nums")
print(reduce(addnums, alist))
```

```
sum of all nums
116
```

In [183]:

```
# filter function check the given conditions is present in sequence and then print the r
# EX :>
list
def edu_age(nums):
    if nums>=4:
        return nums
```

In [184]:

```
agelist = [3, 13, 11, 6, 2.3, 5]
```

In [185]:

```
result_filterobj = filter(edu_age, agelist)
```

In [188]:

```
print(result_filterobj)
print("age for education starts", list(result_filterobj))
```

```
<filter object at 0x0000027B34DEE230>
age for education starts [13, 11, 6, 5]
```

11. Using pen & Paper write the internal mechanism for sum operation using reduce function on this given (Attach paper image for this answer) in doc or colab notebook

Q-11 Write the internal mechanism of sum operation using reduce function on the given list : [47, 11, 42, 13]

#)

```
from functools import reduce
# list of numbers
num = [47, 11, 42, 13]
result = reduce(lambda x, y: x+y, numbers)

print result
```

113

In [1]:

```
## PRACTICAL QUESTIONS
```

In [22]:

```
#  1. Write a Python function that takes a list of numbers as input and returns the sum

#using lambda/filter:

alist = [1, 4, 3, 4, 23, 12, 45]
sum(filter(lambda x: not x%2, alist))
```

Out[22]:

```
20
```

In [23]:

```
# using loop

alist = [2, 4, 23, 9, 8]
sum = 0
for i in alist:
    if i%2 == 0:
        sum += i
```

In [24]:

```
sum
```

Out[24]:

```
14
```

In [39]:

```
# using function
def even_num(a):
    sum_n=0
    for value in a:
        if i %2==0:
            sum_n=sum_n+value
    return sum_n
```

In [40]:

```
print(even_num([2, 4, 3, 6, 10]))
```

In [41]:

```
# 2. Create a Python function that accepts a string and returns the reverse of that stri

def a_function(n):
    return n[ ::-1]
```

In [42]:

```
text = a_function("this is pyhton")
```

In [43]:

```
print(text)
```

```
nothyp si siht
```

In [62]:

```
#  3. Implement a Python function that takes a list of integers and returns a new list c

def values():
    l = list()
```

In [63]:

```
l = list()
for i in range(1, 15):
      l.append(i**2)
```

In [69]:

```
print(l)
values()
```

```
[1, 4, 9, 16, 25, 36, 49, 64, 81, 100, 121, 144, 169, 196]
```

In [ ]:

```
# 4. Write a Python function that checks if a given number is prime or not from 1 to 200
```

In [89]:

```
import math
```

In [93]:

```
def primecheck(x):
    status = 1
    for i in range(2, int(math.sqrt(x)) + 1):
        if x % i == 0:
            status = 0
            print("not prime")
            break
        else:
            continue
        if status == 1:
            print("prime")
        return status
num = int(input("enter the number:"))
ret = primecheck(num)
```

```
not prime
```

In [98]:

```
# 5. Create an iterator class in Python that generates the Fibonacci sequence up to a sp
```

In [129]:

```
class fibonaciIterator:
    def __init__(self, limit):
        self.limit = limit
        self.a = 0
        self.b = 1

    def __iter__(self):
        return self
    def __next__(self):
        if self.a> self.limit:
            raise StopIteration
        current = self.a
        self.a, self.b = self.b, self.a + self.b
        return current
fib_iterator = fibonaciIterator(27)
print("using loop:")
for num in fib_iterator:
    print(num)
```

```
using loop:
0
1
1
2
3
5
8
13
21
```

In [132]:

```
# 6. Write a generator function in Python that yields the powers of 2 up to a given expo

import math
def power_generator(base, exponent):
    result = 1
    for i in range(exponent + 1):
        yield result
        result *= base

base = int(input("base no"))
exponent = int(input("exp no"))
power_gen = power_generator(base, exponent)
print(f"power of {base} up to exponent {exponent}:")
for power in power_gen:
    print(power)
```

```
power of 2 up to exponent 2:
1
2
4
```

In [207]:

```
#  7. Implement a generator function that reads a file line by line and yields each line
def data_from_file(fname, sep=";"):
    with open(fname) as file_iter:
        for line in file_iter:
            line = line.strip()
```

```
            if not line:
                continue
             try:
                id, type_, leg, time = line.split(sep)
            except valueERROR:
            try:
                leg = int(leg):
            except valueERROR:
                leg = "none"
            yield data(type_, leg, int(time) ,id)
```

```
  File <string>:8
    try:
        ^
IndentationError: unindent does not match any outer indentation level
```

In [134]:

```
#  8. Use a lambda function in Python to sort a list of tuples based on the second eleme
subject_marks = [('english', 79), ('social science', 86), ('maths', 92)]
print("original list of tuples:")
print(subject_marks)
```

```
original list of tuples:
[('english', 79), ('social science', 86), ('maths', 92)]
```

In [135]:

```
subject_marks = [('english', 79), ('social science', 86), ('maths', 92)]
print("original list of tuples:")
print(subject_marks)
subject_marks.sort(key=lambda x: x[1])
print("\nsorting the list of tuples:")
print(subject_marks)
```

```
original list of tuples:
[('english', 79), ('social science', 86), ('maths', 92)]

sorting the list of tuples:
[('english', 79), ('social science', 86), ('maths', 92)]
```

In [156]:

```
# 9. Write a Python program that uses `map()` to convert a list of temperatures from Cel
```

In [158]:

```
F = (9/5)*c + 32
c = int(input("enter the temp in celsious:"))
print(str(c)+ "degree celsious is equal to " +str(F)+ "deg far")
```

```
45degree celsious is equal to 113.0deg far
```

In [167]:

```
# 10. Create a Python program that uses `filter()` to remove all the vowels from a given
```

In [168]:

```
def remove_vowels(input_str):
    vowels = "aeiou"
    result = ''
    for char in input_str:
        if char not in vowels:
            result += char
    return result
```

In [169]:

```
input_str = " this is a python assignment "
print(remove_vowels(input_str))
```

```
 ths s  pythn ssgnmnt
```

11) Imagine an accounting routine used in a book shop. It works on a list with sublists, which look like this: Write a Python program, which returns a list with 2-tuples. Each tuple consists of the order number and the product of the price per item and the quantity. The product should be increased by 10,- € if the value of the order is smaller than 100,00 €. Write a Python program using lambda and map

In [208]:

```
# List of oredrs:[order number, Book title, Quantity, Price per item]
orders = [[34587, "learning python, Mark lutz", 4 ,40.95], [98762, "head first pyhton, P
result = list(map(lambda x: (x[0], x[2] * x[3] + (10 if x[2] * x[3] < 100 else 0)), orde
```

In [209]:

```
print((result))
```

```
[(34587, 163.8), (98762, 108.85000000000001), (88112, 84.97)]
```

In [ ]:

In [ ]:

In [ ]:

In [ ]: