Q1. Why do we call Python as a general purpose and high-level programming language?

A-1-Python is considered a general-purpose programming language because it can be used for a wide
range of applications, including web development, data analysis,scientific computing, artificial
intelligence, machine learning, game development, and more. It is not limited to a specific domain
or application area, and it provides a lot of flexibility in terms of the tasks it can accomplish.
Python is also a high-level programming language, which means that it abstracts away many of the
low-level details of the computer’s hardware and operating system.
This makes it easier to write code that is more concise, readable, and easier to maintain compared
to lowlevel programming languages like Assembly or C. Python’s syntax is simple and easy to
understand, which reduces the time and effort required to write code and makes it more accessible
to beginners.

Q2. Why is Python called a dynamically typed language?

A-2- Python is called a dynamically typed language because the data type of a variable is determined
by the value it contains when it is created or assigned a value,and it can be changed later if needed.
For example, in Python, you can assign a string value to a variable and then later assign an integer
value to the same variable without having to explicitly specify the data type of the variable. This
is because Python automatically determines the data type based on the value assigned to the
variable.

Q3. List some pros and cons of Python programming language?

A-3- Pros:
*Easy to learn and use: Python has a simple syntax and a large standard library, making it easy to
learn and use for beginners.
*Versatile: Python can be used for a wide range of applications, from web development to scientific
computing, artificial intelligence, machine learning, and more.
*Dynamically typed: Python is a dynamically typed language, which means that the data type of a
variable is determined at runtime, making it more flexible and easier to use.
*Interpreted language: Python is an interpreted language, which means that the code

can be run without
the need for a separate compilation step, making it quicker to test and debug code.
*Large community and support: Python has a large and active community of developers and users,
providing a wealth of resources, libraries, and frameworks to help solve problems and improve
productivity.

Cons:
*Slower execution speed: Python is an interpreted language, which can result in slower execution speeds
compared to compiled languages like C or Java.
*GIL limitation: Python’s Global Interpreter Lock (GIL) can limit performance when running CPU-bound
tasks, as it prevents multiple threads from executing Python code simultaneously.
*Weak on mobile and game development: Python is not as well-suited for mobile app development or
game development compared to other programming languages like Swift,Java, or C++.
*Design restrictions: Python’s design philosophy enforces some restrictions on the code style and
organization, which can limit the programmer’s creativity and flexibility.
*Memory management: Python’s memory management can sometimes lead to issues with memory leaks
and other performance issues if not managed carefully.

Q4. In what all domains can we use Python?

A-4-Python is a versatile programming language that can be used in a wide range of domains and
applications. Here are some of the main domains in which Python is commonly used:

*Web development: Python has a range of web frameworks such as Django, Flask, and Pyramid that
enable developers to build high-performance web applications with ease.
*Data science and machine learning: Python is a popular language for data science and machine
learning tasks, as it provides powerful libraries such as NumPy, Pandas and Scikit-learn, that
simplify data analysis, visualization, and model building.
*Artificial intelligence and deep learning: Python has a rich set of libraries for artificial
intelligence and deep learning, such as TensorFlow, PyTorch, and Keras,that enable developers
to create complex models for image and speech recognition, natural language processing, and more.
*Scientific computing: Python is a popular choice for scientific computing tasks, as it provides
libraries such as SciPy, Matplotlib, and SymPy that enable scientists and researchers to perform

numerical calculations, data analysis, and simulations.
*Game development: Python can be used to develop 2D and 3D games with the help of libraries like
Pygame,Panda3D, and PyOgre.
*Desktop applications: Python can be used to build cross-platform desktop applications using
frameworks such as PyQt, wxPython, and PyGTK.
*DevOps: Python is widely used for automation, scripting, and configuration management tasks in
DevOps,with popular tools such as Ansible, Fabric, and Salt.

Q5. What are variable and how can we declare them?

A-5- A variable is a named reference to a value that can be used throughout the program. A variable
allows you to store data of different types, such as strings,
integers, floating-point numbers, and more.To declare a variable in Python, you simply choose a name for
the variable and use the equal sign "=" to assign a value to it.

Q6. How can we take an input from the user in Python?

A-6-We can take the input from user by syntax:

```
a=input('ENTER THE INPUT')
print('YOU HAVE TYPE: ',a)
```

Q7. What is the default datatype of the value that has been taken as an input using input() function?

A-7- The default data type of the value returned by the input() function in Python is a string.

Q8. What is type casting?

A-8- type casting is the process of converting a value of one data type to another data type.

Q9. Can we take more than one input from the user using single input() function? If yes, how? If no, why?

A-9- Yes we can take multiple inputs from the user using a single input() function call by asking the user
to enter multiple values separated by a delimiter,such as a space or a comma, and

then splitting the input
string into individual values using the split() function.

Q10. What are keywords?

A-10-Keywords in Python are reserved words that have special meaning and purpose in the language.
These words are used to define the syntax and structure of the language itself and cannot be used as
identifiers (i.e., variable names, function names, etc.) in the program.
EX- and,as,assert,break,class,continue,def,del.

Q11. Can we use keywords as a variable? Support your answer with reason.

A-11-No, we cannot use keywords as a variable name in Python. Keywords are reserved words that have
a predefined meaning and usage in the Python language. If we try to use a keyword as a variable
name or any other identifier in our program, we will get a syntax error.

Q12. What is indentation? What's the use of indentaion in Python?

A-12-In Python, indentation refers to the whitespace (spaces or tabs) at the beginning of a line of code. In
dentation is used to define the block of code, i.e. a group of statements that belong together.
The use of indentation in Python is significant because it defines the structure of the code and the logical
flow of the program. Unlike other programming languages,where the braces or brackets are used to group the
code, Python uses indentation for this purpose.

Q13. How can we throw some output in Python?

A-13-In Python, we can use the print() function to display output on the console. The print() function is a
built-in function that takes one or more arguments,formats them as strings, and displays them on the console.

Q14. What are operators in Python?

A-14-Operators in Python are special symbols or keywords that performs operations

on one or more operands
(variables, constants, expressions). Python supports a wide range of operators that can be used for arithmetic,
comparison, logical, and bitwise operations.

Q15. What is difference between / and // operators?

A-15-In Python, the / operator performs floating-point division, while the // operator performs integer
division (also known as floor division).

Q16. Write a code that gives following as an output
iNeuroniNeuroniNeuroniNeuron

A-16-print("iNeuroniNeuroniNeuroniNeuron")

Q17. Write a code to take a number as an input from the user and check if the number is odd or even.

A-17-
```
a=int(input("ENTER YOUR NUMBER: "))
if a%2==0:
print(’the number is even’)
else:
print(’the number is odd’)
```

Q18. What are boolean operator?

A-18-Boolean operators are operators that operate on Boolean values (True and False) and return a Bool
ean result. Python provides three Boolean operators: and, or,
and not.

Q19. What will the output of the following?

```
1 or 0

0 and 0

True and False and True

1 or 0 or 0
```

```
A-19-
1 or 0 : 1
0 and 0 : 0
True and False and True : False
1 or 0 or 0 : 1
```

Q20. What are conditional statements in Python?

A-20-Conditional statements in Python are used to execute different code blocks based on whether a
particular condition is True or False. The two main types of conditional statements in Python are
the if statement and the if-else statement.

Q21. What is use of 'if', 'elif' and 'else' keywords?

A-21-*if statement: The if statement is used to execute a block of code if a particular condition is True.
*if-else statement: The if-else statement is used to execute one of two code blocks depending on whether
a particular condition is True or False.
*elif statement: The elif statements to test multiple conditions.

Q22. Write a code to take the age of person as an input and if age >= 18 display "I can vote". If age is < 18 display "I can't vote".

A-22-
```
a=int(input("enter the age of a person: "))
if a>=18:
print('I can vote')
elif a<18:
print("I can't vote")
```

Q23. Write a code that displays the sum of all the even numbers from the given list.

```
numbers = [12, 75, 150, 180, 145, 525, 50]
```

A-23-
```
sum=0
numbers = [12, 75, 150, 180, 145, 525, 50]
for i in numbers:
if i%2==0:
sum=sum+i
```

```
print(sum)
```

Q24. Write a code to take 3 numbers as an input from the user and display the greatest no as output.

A-24-

```
a=int(input("ENTER FIRST NUMBER: "))
b=int(input("ENTER SECOND NUMBER: "))
c=int(input("ENTER THIRD NUMBER: "))
if a>b and a>c:
print(a,"is greatest")
elif b>a and b>c:
print(b,"is greatest")
elif c>a and c>b:
print(c,"is greatest")
```

Q25. Write a program to display only those numbers from a list that satisfy the following conditions

The number must be divisible by five

If the number is greater than 150, then skip it and move to the next number

If the number is greater than 500, then stop the loop

numbers = [12, 75, 150, 180, 145, 525, 50]

A-25-

```
numbers = [12, 75, 150, 180, 145, 525,50]
lst = []
for num in numbers:
  if num > 150:
    if num > 500:
      break
  elif num%5==0:
    lst.append(num)

print(lst)
```

Q26. What is a string? How can we declare string in Python?

A-26-In python, a string is a sequence of characters. Strings are commonly used to represent text, such
as words and sentences.we can declare a string by enclosing the sequence of

characters within either
single quotes ('...') or double quotes ("...").for ex- str='hello world'.

Q27. How can we access the string using its index?

```
A-27-my_string = "Hello, world!"
first_character = my_string[0]  # Access the first character (index 0)
second_character = my_string[1]  # Access the second character (index 1)
last_character = my_string[-1]  # Access the last character (index -1)
second_last_character = my_string[-2] # Access the second-last character (index -2)
```

Q28. Write a code to get the desired output of the following

```
string = "Big Data iNeuron"
desired_output = "iNeuron"
```

A-28-print(string[9:16:1])

Q29. Write a code to get the desired output of the following

```
string = "Big Data iNeuron"
desired_output = "norueNi"
```

A-29-print(string[-1:-8:-1])

Q30. Resverse the string given in the above question.

A-30-print(string[::-1])

Q31. How can you delete entire string at once?

A-31-We acn delete the string using the delete function i.e del string.

Q32. What is escape sequence?

A-32-In Python, an escape sequence is a sequence of characters that represent a
special
character or a control character within a string.for ex- \n: newline,\t: tab,\':
single

quote ('),\": double quote ("),\\: backslash ().

Q33. How can you print the below string?

'iNeuron's Big Data Course'

A-33-print("'iNeuron\'s Big Data Course'")

Q34. What is a list in Python?

A-34-In Python, a list is a collection of items, where each item can be of any data type,
such as integers, floats, strings, or even other lists. Lists are ordered, meaning that
the order in which items are added to a list is preserved, and can be accessed using an
index.for ex:my_list = [1, 2, "three", 4.0, [5, 6, 7]].

Q35. How can you create a list in Python?

A-35-To create a list in Python, we can enclose a comma-separated sequence of items
within square brackets ([]). For example, to create a list of integers.
for ex:my_list = [1, 2, "three", 4.0, [5, 6, 7]].

Q36. How can we access the elements in a list?

A-36-In Python, you can access individual elements in a list by using their index. The
index of the first item in a list is 0, and the index of the last item is len(list)

- 1 (where list is the name of the list variable).
for ex:my_list = [1, 2, 3, 4, 5]
first_item=my_list[0].

Q37. Write a code to access the word "iNeuron" from the given list.

lst = [1,2,3,"Hi",[45,54, "iNeuron"], "Big Data"]

A-37-print((lst[4])[2])

Q38. Take a list as an input from the user and find the length of the list.

A-38-

```
l1=input('enter your list seperated by comma :').split(',')
print(l1)
print(len(l1))
```

Q39. Add the word "Big" in the 3rd index of the given list.

lst = ["Welcome", "to", "Data", "course"]

A-39-

```
lst = ["Welcome", "to", "Data", "course"]
lst.insert(3,'Big')
print(lst)
```

Q40. What is a tuple? How is it different from list?

A-40-In Python, a tuple is an ordered, immutable sequence of elements. Like a list, a
tuple can contain elements of different data types, such as integers, strings, and
other objects. However, unlike a list, a tuple cannot be modified once it is
created.
One major difference between tuples and lists is that tuples are immutable, which means
that their contents cannot be changed once they are created. This means that you can
not add, remove, or modify elements in a tuple like you can with a list.
Another difference is that tuples are usually used to represent fixed collections
of data, while lists are used for collections that may change over time.

Q41. How can you create a tuple in Python?

A-41-In Python, you can create a tuple by enclosing a sequence of elements in
parentheses, separated by commas.for ex:my_tuple = (1, 2, 3, "four", 5.0).

Q42. Create a tuple and try to add your name in the tuple. Are you able to do it?
Support your answer with reason.

A-42-No we cannot do it as tuples are immutable, so we cannot add or modify their
contents once they are created. However, we can create a new tuple that includes
name by concatenating two tuples together.

Q43. Can two tuple be appended. If yes, write a code for it. If not, why?

A-43-Yes, we can append two tuples in Python to create a new tuple that contains all the elements from both tuples.

```
tuple1 = (1, 2, 3)
tuple2 = (4, 5, 6)
new_tuple = tuple1 + tuple2
print(new_tuple)
```

Q44. Take a tuple as an input and print the count of elements in it.

A-44-

```
T1=tuple(input('ENTER THE TUPLE SEPERATED BY COMMA :').split(','))
print(T1,'length of tuple is :',len(T1))
```

Q45. What are sets in Python?

A-45- In Python, a set is a collection of unique elements. The elements can be of any data type and are enclosed in curly braces {}. Sets are unordered, so the elements in a set are not stored in any particular order.
For ex-my_set = {1, 2, 3, 4, 5}

Q46. How can you create a set?

A-46- You can create a set in Python by enclosing a comma-separated sequence of values in curly braces {} or You can also create an empty set by calling the set() function or you can create a set from a list or other iterable object by passing it to the set() function,.for ex-my_set = {1, 2, 3, 4, 5} or empty_set = set() or my_list = [1, 2, 3, 4, 5] my_set = set(my_list).

Q47. Create a set and add "iNeuron" in your set.

A-47-

```
S1={1,2,3}
S1.add('iNeuron')
print(S1)
```

Q48. Try to add multiple values using add() function.

A-48-

```
S1={1,2,3}
S1.update({'iNeuron',23,'delta'})
print(S1)
```

Q49. How is update() different from add()?

A-49- add() takes only single argument as input whereas update() takes multiple arguments as input either list,tuple or a set.
for ex-

```
S1={1,2,3}                    S1={1,2,3}
S1.add('iNeuron')             S1.update({'iNeuron',23,'delta'})
print(S1)                     print(S1)
```

Q50. What is clear() in sets?

A-50-In Python sets, clear() is a built-in function that removes all the elements from a set. After the clear() function is called, the set becomes empty.
for ex

```
my_set = {1, 2, 3, 4, 5}
my_set.clear()
print(my_set)
```

Q51. What is frozen set?

A-51-In Python, a frozen set is a special type of set that is immutable, meaning that its elements cannot be changed once it has been created.
In other words, a frozen set is a read-only set.
Frozen sets are created using the frozenset() function, which takes an iterable object (such as a list or tuple) as its argument and returns a new frozen set object.

for ex-

```
my_frozen_set = frozenset([1, 2, 3])
print(my_frozen_set)
print(1 in my_frozen_set)
print(len(my_frozen_set))
```

Q52. How is frozen set different from set?

A-52-The main difference between a frozen set and a regular set in Python is that a frozen set is immutable (read-only), while a regular set is mutable (can be modified).

Q53. What is union() in sets? Explain via code.

A-53- In Python sets, union() is a built-in function that returns a new set containing all the unique elements from two or more sets. The union() function does not modify the original sets, but instead returns a new set that contains the union of the original sets.

for ex-
```
set1 = {1, 2, 3}
set2 = {2, 3, 4}
set3 = {3, 4, 5}
union_set = set1.union(set2, set3)
print(union_set)
```

Q54. What is intersection() in sets? Explain via code.

A-54- In Python sets, intersection() is a built-in function that returns a new set containing only the elements that are common to two or more sets. The intersection() function does not modify the original sets, but instead returns a new set that contains the intersection of the original sets.

for ex-
```
set1 = {1, 2, 3}
set2 = {2, 3, 4}
set3 = {3, 4, 5}
intersection_set = set1.intersection(set2, set3)
print(intersection_set)
```

Q55. What is dictionary in Python?

A-55-In Python, a dictionary is a built-in data structure that represents a collection of key-value pairs.

for ex-
```
my_dict = {'apple': 2, 'banana': 3, 'orange': 1}
print(my_dict)
```

Q56. How is dictionary different from all other data structures.

A-56- some of the key differences between dictionaries and other data structures:

*Dictionaries are unordered: Unlike lists and tuples, dictionaries

do not have an inherent order.
*Dictionaries are mutable: Dictionaries can be modified by adding, updating or deleting key-value pairs. This is different from tuples.
*Dictionaries use keys to access values: Unlike lists and tuples, which use integer indices to access elements, dictionaries use keys to access values.

Q57. How can we delare a dictionary in Python?

A-57- In Python, you can declare a dictionary using curly braces {} or by using the dict() constructor.
for ex-

```
my_dict = {'name': 'Alice', 'age': 30, 'city': 'New York'}
print(my_dict)
```

Q58. What will the output of the following?

```
var = {}
print(type(var))
```

A-58-<class 'dict'>

Q59. How can we add an element in a dictionary?

A-59-

```
my_dict = {'name': 'John', 'age': 35}
my_dict['email'] = 'john@example.com'
print(my_dict)
```

Q60. Create a dictionary and access all the values in that dictionary.

A-60-

```
D={'rishabh':26,'prince':25,'karthik':28}
for value in D.values():
  print(value)
```

Q61. Create a nested dictionary and access all the element in the inner dictionary.

A-61-

```
D={'person1':{'name':'rishabh','age':26},'person2':{'name':
'karthik','age':28}}
```

```
for K1,V1 in D.items():
  print(f"{K1}:")
  for K2,V2 in V1.items():
    print(f"{K2} :{V2}")
```

Q62. What is the use of get() function?

A-62-The get() function is a method in Python dictionaries that is used to retrieve the value of a specified key. The get() function takes one required argument, which is the key to be searched for in the dictionary, and an optional second argument, which is the default value to be returned if the key is not found in the dictionary.

for ex-

```
my_dict = {'name': 'Alice', 'age': 30}
name = my_dict.get('name')
gender = my_dict.get('gender', 'Unknown')
print(name)
print(gender)
```

Q63. What is the use of items() function?

A-63-In Python, the items() function is a method that is used to return a list of key-value pairs (or items) of a dictionary. It returns a view object that contains the key-value pairs of the dictionary as tuples.

for ex-

```
my_dict = {'name': 'Alice', 'age': 30}
items = my_dict.items()
print(items)
for key, value in items:
    print(key, value)
```

Q64. What is the use of pop() function?

A-64-In Python, the pop() function is a method that is used to remove an item with the specified key from a dictionary and return its value. The pop() method takes one required argument, which is the key to be removed from the dictionary.

for ex-

```
my_dict = {'name': 'Alice', 'age': 30}
age = my_dict.pop('age')
gender = my_dict.pop('gender', 'Unknown')
print(age)
print(my_dict)
print(gender)
```

Q65. What is the use of popitems() function?

A-65-In Python, the popitem() function is a method that is used to remove and return
an arbitrary (key, value) item from a dictionary. The popitem() method does not take
any arguments.

for ex-
```
my_dict = {'name': 'Alice', 'age': 30}
item = my_dict.popitem()
print(item)
print(my_dict)
```

Q66. What is the use of keys() function?

A-66-In Python, the keys() function is a method that is used to get a list of all the
keys in a dictionary. The keys() method does not take any arguments.

for ex-
```
my_dict = {'name': 'Alice', 'age': 30}
keys = my_dict.keys()
print(keys)
```

Q67. What is the use of values() function?

A-67-In Python, the values() function is a method that is used to get a list of all the
values in a dictionary. The values() method does not take any arguments.

for ex-
```
my_dict = {'name': 'Alice', 'age': 30}
values = my_dict.values()
print(values)
```

Q68. What are loops in Python?

A-68-In Python, loops are used to execute a block of code repeatedly until a certain condition is met. There are two types of loops in Python: for loops and while loops.

Q69. How many type of loop are there in Python?

A-69-There are two types of loops in Python: for loops and while loops.

Q70. What is the difference between for and while loops?

A-70-The main difference between for loops and while loops is in how they iterate over a
sequence. A for loop is used to iterate over a sequence (such as a list, tuple, or string)
and execute the block of code for each item in the sequence. A while loop, on the other
hand, is used to repeatedly execute a block of code as long as a certain condition is true.

Q71. What is the use of continue statement?

A-71-In Python, the continue statement is used to skip the current iteration of a loop
(for loop or while loop) and continue with the next iteration.

Q72. What is the use of break statement?

A-72-In Python, the break statement is used to immediately exit out of a loop
(for loop or while loop) when a certain condition is met.

Q73. What is the use of pass statement?

A-73-In Python, the pass statement is a null operation that does nothing. It is used as a
placeholder statement when a statement is required syntactically but you don't want to
execute any code.

Q74. What is the use of range() function?

A-74-The range() function can be useful in situations where you need to generate a sequence
of numbers to control the number of times a loop is executed or to perform some other
operation that requires a sequence of numbers.

Q75. How can you loop over a dictionary?

A-75-

```
my_dict = {'apple': 2, 'banana': 3, 'orange': 4}
for key in my_dict:
    print(key, my_dict[key])
```

Coding problems
Q76. Write a Python program to find the factorial of a given number.

A-76-
```
def factorial(n):
  if n==0 or n==1:
    return 1
  else:
    return n*factorial(n-1)

n=int(input('ENTER YOUR NUMBER :'))
if n<0:
  print('INVALID INPUT')
else:
  print('the factorial of number is :',factorial(n))
```

Q77. Write a Python program to calculate the simple interest. Formula to calculate simple interest is SI = (PRT)/100

A-77-
```
p=int(input('enter the principal amount :'))
r=float(input("enter the intrest rate :"))
t=int(input('enter the time period :'))
si=(p*r*t)/100
print('the simple intrest is :',si)
```

Q78. Write a Python program to calculate the compound interest. Formula of compound interest is A = P(1+ R/100)^t.

A-78-
```
p=int(input('enter the principal amount :'))
r=float(input("enter the intrest rate :"))
t=int(input('enter the time period :'))
a=p*(1+(r/100))**t
ci=a-p
print('the compound intrest is :',ci)
```

Q79. Write a Python program to check if a number is prime or not.

A-79-
```
n=int(input('enter the number :'))
if n<2:
  print(n,' is not a prime number')
else:
  is_prime=True
  for i in range (2,int(n/2)+1):
    if n%i==0:
      is_prime=False
      break
  if is_prime:
    print(n,' is a prime number')
  else:
    print(n,' is not a prime number')
```

Q80. Write a Python program to check Armstrong Number.

A-80-
```
n=int(input('enter the number :'))
no_of_digits=len(str(n))
sum_of_cubes=sum(int(digit)**no_of_digits for digit in str(n))
if n==sum_of_cubes:
  print(n,' is a armstrong number')
else:
  print(n,' is not a armstrong number')
```

Q81. Write a Python program to find the n-th Fibonacci Number.

A-81-
```
def fibonacci(n):
  if n<=0:
    print('INVALID INPUT')
  elif n==1:
    return 0
  elif n==2:
    return 1
  else:
    return fibonacci(n-1)+fibonacci(n-2)

n=int(input('enter your number :'))

print("the required fibonacci series will be :",fibonacci(n))
```

Q82. Write a Python program to interchange the first and last element in a list.

A-82-

```
def interchange_first_last(lst):
  if len(lst)<1:
    return None
  else:
    lst[0],lst[-1]=lst[-1],lst[0]
    return lst
lst=[1,2,3,4,5]
print(lst)
swap_lst=interchange_first_last(lst)
print(swap_lst)
```

Q83. Write a Python program to swap two elements in a list.

A-83-

```
def swap_elements(lst,index1,index2):
  if index1<0 or index1>len(lst) or index2<0 or index2>len(lst):
    return None
  else:
    lst[index1],lst[index2]=lst[index2],lst[index1]
    return lst
lst=[1,2,3,4,5]
print(lst)
swap_lst=swap_elements(lst,1,3)
print(swap_lst)
```

Q84. Write a Python program to find N largest element from a list.

A-84-

```
def n_largest_element(lst,n):
  sorted_lst=sorted(lst,reverse=True)
  return sorted_lst[:n]

lst=[1,8,23,9,4,5]
n=3
print(f"{n} largest element in the list {lst} are : {n_largest_element(lst,n)}")
```

Q85. Write a Python program to find cumulative sum of a list.

A-85-

```
def cummulative_sum(lst):
  cum_sum=[]
  total=0
  for num in lst:
    total =total+num
    cum_sum.append(total)
  return cum_sum

l=[1,2,3,4,5]
print(f"the cummulative sum of the {l} is :{cummulative_sum(l)}")
```

Q86. Write a Python program to check if a string is palindrome or not.

A-86-

```
def palindrome(S):
  S=S.lower().replace(" ","")
  return S==S[::-1]

S1="A man a plan a canal Panama"
print(palindrome(S1))
```

Q87. Write a Python program to remove i'th element from a string.

A-87-

```
def remove_ith_char(s, i):
  s_list = list(s)
  del s_list[i]
  return ''.join(s_list)

S1='HELLO WORLD'
print(remove_ith_char(S1,6))
```

Q88. Write a Python program to check if a substring is present in a given string.

A-88-
```
def is_substring(s,sub):
  if sub in s:
    return True
  else:
    return False


string='hello world'
substring='world'
print(is_substring(string,substring))
```

Q89. Write a Python program to find words which are greater than given length k.

A-89-
```
def find_long_words(s,k):
  words=s.split()
  long_words=[word for word in words if len(word)>k]
  return long_words

my_string = "The quick brown fox jumps over the lazy dog"
long_words = find_long_words(my_string, 4)
print('the required list is :',long_words)
```

Q90. Write a Python program to extract unquire dictionary values.

A-90-
```
d= {'rishabh': [5, 6, 7, 8],'is': [10, 11, 7, 5],'a': [6, 12, 10, 8],'good': [1, 2,
5],'boy': [2, 7, 12, 9]}
l1=list(sorted({e for value in d.values() for e in value}))
print(l1)
```

Q91. Write a Python program to merge two dictionary.

A-91-
```
def merge_dict(d1,d2):
  merged_dict=d1.copy()
  merged_dict.update(d2)
  return merged_dict

dict1 = {'a': 10, 'b': 8}
dict2 = {'d': 6, 'c': 4}
d=merge_dict(dict1,dict2)
print(d)
```

Q92. Write a Python program to convert a list of tuples into dictionary.

```
Input : [('Sachin', 10), ('MSD', 7), ('Kohli', 18), ('Rohit', 45)]
Output : {'Sachin': 10, 'MSD': 7, 'Kohli': 18, 'Rohit': 45}
```

A-92-
```
print(dict([('Sachin', 10), ('MSD', 7), ('Kohli', 18), ('Rohit', 45)]))
```

Q93. Write a Python program to create a list of tuples from given list having number and its cube in each tuple.

```
Input: list = [9, 5, 6]
Output: [(9, 729), (5, 125), (6, 216)]
```

A-93-
```
list1 = [9, 5, 6]
l= [(val, pow(val, 3)) for val in list1]
print(l)
```

Q94. Write a Python program to get all combinations of 2 tuples.

```
Input : test_tuple1 = (7, 2), test_tuple2 = (7, 8)
Output : [(7, 7), (7, 8), (2, 7), (2, 8), (7, 7), (7, 2), (8, 7), (8, 2)]
```

A-94-
```
test_tuple1 = (7, 2)
test_tuple2 = (7, 8)
l = [(a, b) for a in test_tuple1 for b in test_tuple2]
l = l + [(a, b) for a in test_tuple2 for b in test_tuple1]
print(l)
```

Q95. Write a Python program to sort a list of tuples by second item.

```
Input : [('for', 24), ('Geeks', 8), ('Geeks', 30)]
Output : [('Geeks', 8), ('for', 24), ('Geeks', 30)]
```

A-95-
```
tuples_list = [('for', 24), ('Geeks', 8), ('Geeks', 30)]
sorted_list = sorted(tuples_list, key=lambda x: x[1])
print(sorted_list)
```

Q96. Write a python program to print below pattern.

```
*
* *
* * *
* * * *
* * * * *
```

A-96-

```
def print_pattern(num_rows):
  for i in range(num_rows):
    for j in range(i+1):
      print("*", end=" ")
    print()
print_pattern(5)
```

Q97. Write a python program to print below pattern.

```
    *
   **
  ***
 ****
*****
```

A-97-

```
def print_pattern(num_rows):
  for i in range(num_rows):
    print(" "*(num_rows-i-1) + "*"*(i+1))

print_pattern(5)
```

Q98. Write a python program to print below pattern.

```
    *
   * *
  * * *
 * * * *
* * * * *
```

A-98-

```
def print_pattern(num_rows):
  for i in range(num_rows):
    print(" "*(num_rows-i-1) + "* "*(i+1))
```

```
print_pattern(5)
```

Q99. Write a python program to print below pattern.

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

A-99-

```
for i in range(1,6):
  for j in range(1,i+1):
    print(j,end=" ")
  print()
```

Q100. Write a python program to print below pattern.

```
A
B B
C C C
D D D D
E E E E E
```

A-100-

```
for i in range(65, 70):
    for j in range(i - 64):
        print(chr(i), end=' ')
    print()
```