**1) Types of Applications**

* **Web Applications**: Run on browsers, e.g., Gmail, Amazon.
* **Desktop Applications**: Installed on computers, e.g., MS Word.
* **Mobile Applications**: Designed for mobile devices, e.g., WhatsApp.
* **Embedded Systems**: Integrated into hardware, e.g., IoT devices.
* **Game Applications**: Developed for entertainment, e.g., PUBG.
* **AI Applications**: Powered by machine learning, e.g., ChatGPT.

**2) What is Programming?**

Programming is the process of designing and implementing algorithms to solve problems using a programming language like Python, Java, or C++.

**3) What is Python?**

Python is a high-level, interpreted programming language known for its simplicity and readability. It supports multiple paradigms like object-oriented, procedural, and functional programming.

**4) Python Program: Check if a Number is Positive, Negative, or Zero**

python

Copy code

number = int(input("Enter a number: "))

if number > 0:

print("Positive")

elif number < 0:

print("Negative")

else:

print("Zero")

**5) Python Program: Factorial of a Given Number**

python

Copy code

def factorial(n):

result = 1

for i in range(1, n + 1):

result \*= i

return result

num = 5

print(f"Factorial of {num}: {factorial(num)}")

**6) Python Program: Fibonacci Series**

python

Copy code

def fibonacci(n):

fib\_series = [0, 1]

for i in range(2, n):

fib\_series.append(fib\_series[-1] + fib\_series[-2])

return fib\_series[:n]

print(f"Fibonacci series: {fibonacci(10)}")

**7) How is Memory Managed in Python?**

* Memory is managed by Python’s **Garbage Collector**.
* Python uses **private heap space** for object storage.
* The **gc module** can manually trigger garbage collection.

**8) Purpose of continue Statement**

The continue statement skips the current iteration of a loop and moves to the next iteration.

Example:

python

Copy code

for i in range(5):

if i == 2:

continue

print(i)

**9) Python Program: Swap Two Numbers**

**With Temp Variable**:

python

Copy code

a, b = 10, 20

temp = a

a = b

b = temp

print(a, b)

**Without Temp Variable**:

python

Copy code

a, b = 10, 20

a, b = b, a

print(a, b)

**10) Python Program: Check Even or Odd**

python

Copy code

number = int(input("Enter a number: "))

if number % 2 == 0:

print("Even")

else:

print("Odd")

**11) Python Program: Test if a Letter is a Vowel**

python

Copy code

letter = input("Enter a letter: ").lower()

if letter in "aeiou":

print("Vowel")

else:

print("Not a vowel")

**12) Python Program: Sum of Three Integers**

python

Copy code

def sum\_of\_three(a, b, c):

if a == b or b == c or a == c:

return 0

return a + b + c

print(sum\_of\_three(1, 2, 2))

**13) Python Program: Return True if Two Numbers Equal, or Sum/Difference is 5**

python

Copy code

def check\_conditions(a, b):

return a == b or abs(a - b) == 5 or a + b == 5

print(check\_conditions(10, 5))

**14) Python Program: Sum of First n Positive Integers**

python

Copy code

def sum\_of\_n(n):

return n \* (n + 1) // 2

print(sum\_of\_n(5))

**15) Python Program: Calculate the Length of a String**

python

Copy code

text = "Hello"

print(len(text))

**16) Python Program: Count Frequency of Characters in a String**

python

Copy code

from collections import Counter

text = "hello"

print(Counter(text))

**17) Negative Indexes in Python**

Negative indexes allow you to access elements from the end of a sequence.  
Example:

python

Copy code

lst = [1, 2, 3]

print(lst[-1]) # Output: 3

**18) Python Program: Count Occurrences of a Substring**

python

Copy code

text = "hello world, hello"

print(text.count("hello"))

**19) Python Program: Count Words in a String**

python

Copy code

text = "hello world hello"

words = text.split()

word\_count = {word: words.count(word) for word in words}

print(word\_count)

**20) Python Program: Swap First Two Characters of Two Strings**

python

Copy code

def swap\_first\_two(str1, str2):

return str2[:2] + str1[2:], str1[:2] + str2[2:]

print(swap\_first\_two("abc", "xyz"))

**21) Python Program: Add 'ing' or 'ly' to a String**

python

Copy code

def add\_suffix(string):

if len(string) < 3:

return string

if string.endswith("ing"):

return string + "ly"

return string + "ing"

print(add\_suffix("play"))

**22) Python Function: Reverse a String if its Length is a Multiple of 4**

python

Copy code

def reverse\_if\_multiple\_of\_4(s):

return s[::-1] if len(s) % 4 == 0 else s

print(reverse\_if\_multiple\_of\_4("hello"))

print(reverse\_if\_multiple\_of\_4("abcd"))

**23) Python Program: Get First and Last 2 Characters of a String**

python

Copy code

def first\_last\_2(s):

return s[:2] + s[-2:] if len(s) >= 2 else ""

print(first\_last\_2("hello"))

**24) Python Function: Insert a String in the Middle of Another String**

python

Copy code

def insert\_middle(s1, s2):

middle = len(s1) // 2

return s1[:middle] + s2 + s1[middle:]

print(insert\_middle("abc", "xyz"))

**25) What is List? How Will You Reverse a List?**

A **list** is a collection of ordered, mutable items, which can be of different data types.

To reverse a list:

python

Copy code

lst = [1, 2, 3, 4]

lst.reverse() # In-place reverse

print(lst)

# Or using slicing

print(lst[::-1])

**26) How Will You Remove the Last Object from a List?**

You can use the pop() method to remove the last item from a list:

python

Copy code

lst = [1, 2, 3, 4]

lst.pop()

print(lst)

**27) List1 = [2, 33, 222, 14, 25], What is list1[-1]?**

list1[-1] returns the last element of the list:

python

Copy code

list1 = [2, 33, 222, 14, 25]

print(list1[-1]) # Output: 25

**28) Difference Between append() and extend() Methods**

* **append()**: Adds a single element to the end of the list.
* **extend()**: Adds multiple elements (from an iterable) to the list.

python

Copy code

lst = [1, 2]

lst.append(3) # lst = [1, 2, 3]

lst.extend([4, 5]) # lst = [1, 2, 3, 4, 5]

**29) Python Function to Get the Largest, Smallest, and Sum of All Numbers in a List**

python

Copy code

def list\_stats(lst):

return {"largest": max(lst), "smallest": min(lst), "sum": sum(lst)}

numbers = [1, 2, 3, 4, 5]

print(list\_stats(numbers))

**30) How Will You Compare Two Lists?**

You can compare two lists using == to check if they are identical:

python

Copy code

list1 = [1, 2, 3]

list2 = [1, 2, 3]

print(list1 == list2) # Output: True

# For element-wise comparison

print([a == b for a, b in zip(list1, list2)]) # Output: [True, True, True]

**31) Python Program to Count Strings Where Length >= 2 and First and Last Characters are Same**

python

Copy code

def count\_same\_start\_end(lst):

return sum(1 for s in lst if len(s) >= 2 and s[0] == s[-1])

strings = ["aba", "abc", "xyx", "abcd"]

print(count\_same\_start\_end(strings))

**32) Python Program to Remove Duplicates from a List**

python

Copy code

lst = [1, 2, 2, 3, 4, 4]

unique\_lst = list(set(lst))

print(unique\_lst)

**33) Python Program to Check if a List is Empty**

python

Copy code

lst = []

print("Empty" if not lst else "Not Empty")

**34) Python Function to Check if Two Lists Have a Common Member**

python

Copy code

def have\_common(lst1, lst2):

return any(item in lst2 for item in lst1)

list1 = [1, 2, 3]

list2 = [3, 4, 5]

print(have\_common(list1, list2))

**35) Python Program to Generate List of First and Last 5 Squared Numbers**

python

Copy code

squares = [x\*\*2 for x in range(1, 31)]

first\_last = squares[:5] + squares[-5:]

print(first\_last)

**36) Python Function to Return a List of Unique Elements**

python

Copy code

def unique\_elements(lst):

return list(set(lst))

lst = [1, 2, 2, 3, 4]

print(unique\_elements(lst))

**37) Python Program to Convert a List of Characters into a String**

python

Copy code

lst = ['h', 'e', 'l', 'l', 'o']

print("".join(lst))

**38) Python Program to Select a Random Item from a List**

python

Copy code

import random

lst = [1, 2, 3, 4, 5]

print(random.choice(lst))

**39) Python Program to Find the Second Smallest Number in a List**

python

Copy code

lst = [2, 3, 1, 4]

sorted\_lst = sorted(lst)

print(sorted\_lst[1]) # Second smallest number

**40) Python Program to Get Unique Values from a List**

python

Copy code

lst = [1, 2, 2, 3, 4]

unique\_values = list(set(lst))

print(unique\_values)

**41) Python Program to Check if a List Contains a Sublist**

python

Copy code

def contains\_sublist(lst, sublist):

return any(lst[i:i+len(sublist)] == sublist for i in range(len(lst) - len(sublist) + 1))

lst = [1, 2, 3, 4, 5]

sublist = [2, 3]

print(contains\_sublist(lst, sublist))

**42) Python Program to Split a List into Different Variables**

python

Copy code

lst = [1, 2, 3, 4]

a, b, c, d = lst

print(a, b, c, d)

**43) What is Tuple? Difference Between List and Tuple**

* A **tuple** is an immutable, ordered collection.
* **List** is mutable, meaning you can change its elements, whereas **Tuple** is immutable.

Example:

python

Copy code

lst = [1, 2, 3]

tup = (1, 2, 3)

**44) Python Program to Create a Tuple with Different Data Types**

python

Copy code

tup = (1, "hello", 3.14, True)

print(tup)

**45) Python Program to Unzip a List of Tuples into Individual Lists**

python

Copy code

tup\_list = [(1, 'a'), (2, 'b'), (3, 'c')]

unzipped = zip(\*tup\_list)

print(list(unzipped))

**46) Python Program to Convert a List of Tuples into a Dictionary**

python

Copy code

tuple\_list = [("a", 1), ("b", 2), ("c", 3)]

dictionary = dict(tuple\_list)

print(dictionary)

**47) How Will You Create a Dictionary Using Tuples in Python?**

python

Copy code

tuple\_list = [("x", 10), ("y", 20)]

dictionary = {key: value for key, value in tuple\_list}

print(dictionary)

**48) Python Script to Sort a Dictionary by Value (Ascending and Descending)**

python

Copy code

def sort\_dict\_by\_value(d, ascending=True):

return dict(sorted(d.items(), key=lambda item: item[1], reverse=not ascending))

my\_dict = {"a": 3, "b": 1, "c": 2}

ascending = sort\_dict\_by\_value(my\_dict, ascending=True)

descending = sort\_dict\_by\_value(my\_dict, ascending=False)

print(f"Ascending order: {ascending}")

print(f"Descending order: {descending}")

**49) Python Script to Concatenate Dictionaries**

python

Copy code

dict1 = {"a": 1, "b": 2}

dict2 = {"c": 3, "d": 4}

dict3 = {"e": 5}

# Concatenating dictionaries

result = {\*\*dict1, \*\*dict2, \*\*dict3}

print(f"Concatenated dictionary: {result}")

**50) Python Script to Check if a Given Key Exists in a Dictionary**

python

Copy code

def check\_key(dictionary, key):

return key in dictionary

my\_dict = {"a": 1, "b": 2, "c": 3}

key\_to\_check = "b"

print(f"Does the key '{key\_to\_check}' exist? {check\_key(my\_dict, key\_to\_check)}")

**51) How Do You Traverse Through a Dictionary Object in Python?**

python

Copy code

my\_dict = {"a": 1, "b": 2, "c": 3}

# Using a loop

for key, value in my\_dict.items():

print(f"{key}: {value}")

**52) How Do You Check the Presence of a Key in a Dictionary?**

python

Copy code

my\_dict = {"a": 1, "b": 2, "c": 3}

print("a" in my\_dict) # Output: True

**53) Python Script to Print a Dictionary with Keys as Numbers Between 1 and 15**

python

Copy code

my\_dict = {i: i \* i for i in range(1, 16)}

print(my\_dict)

**54) Python Program to Check Multiple Keys Exist in a Dictionary**

python

Copy code

my\_dict = {"a": 1, "b": 2, "c": 3}

keys = ["a", "c"]

print(all(k in my\_dict for k in keys))

**55) Python Script to Merge Two Dictionaries**

python

Copy code

dict1 = {"a": 1}

dict2 = {"b": 2}

merged\_dict = {\*\*dict1, \*\*dict2}

print(merged\_dict)

**56) Python Program to Map Two Lists into a Dictionary**

python

Copy code

keys = ["a", "b", "c"]

values = [1, 2, 3]

result = dict(zip(keys, values))

print(result)