Practice Questions

1. Print all even numbers from 1 to 50

```
for i in range(1, 51):
    if i % 2 == 0:
        print(i)
```

2. Find the largest of three numbers

```
a = int(input("Enter first: "))
b = int(input("Enter second: "))
c = int(input("Enter third: "))

largest = max(a, b, c)
print("Largest is:", largest)
```

3. Check if a number is prime

```
num = int(input("Enter number: "))
if num < 2:
    print("Not Prime")
else:
    for i in range(2, num):
        if num % i == 0:
            print("Not Prime")
            break
    else:
        print("Prime")
```

4. Count digits in a number

```
num = int(input("Enter number: "))
count = 0

while num > 0:
    count += 1
    num //= 10

print("Total digits:", count)
```

5. Reverse a number

```
num = int(input("Enter number: "))
rev = 0

while num > 0:
    digit = num % 10
    rev = rev * 10 + digit
    num //= 10

print("Reversed number:", rev)
```

6. Multiplication table of given number

```
n = int(input("Enter number: "))

for i in range(1, 11):
    print(n, "x", i, "=", n * i)
```

7. Sum of first N natural numbers

```
N = int(input("Enter N: "))
total = N * (N + 1) // 2
print("Sum =", total)
```

8. Check vowel or consonant

```
ch = input("Enter character: ").lower()

if ch in "aeiou":
    print("Vowel")
else:
    print("Consonant")
```

9. Find highest and lowest in a list

```
lst = [12, 45, 3, 67, 22]

print("Highest:", max(lst))
print("Lowest:", min(lst))
```

10. Check if a string is palindrome

```
s = input("Enter string: ")

if s == s[::-1]:
    print("Palindrome")
else:
    print("Not Palindrome")
```

11. Count vowels in a string

```
s = input("Enter string: ").lower()
count = 0

for ch in s:
    if ch in "aeiou":
        count += 1

print("Total vowels:", count)
```

12. Find the second largest number in a list

```
lst = [10, 50, 20, 40, 30]

lst = list(set(lst))   # remove duplicates
lst.sort()

print("Second largest:", lst[-2])
```

13. Remove duplicates from a list

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

unique = list(set(lst))
print(unique)
```

14. Sum of digits of a number

```
num = int(input("Enter number: "))
total = 0

while num > 0:
    total += num % 10
    num //= 10

print("Sum of digits:", total)
```

15. Frequency of each character in a string

```
s = input("Enter string: ")

freq = {}
for ch in s:
    freq[ch] = freq.get(ch, 0) + 1

print(freq)
```

16. Factorial using a function

```
def factorial(n):
    fact = 1
    for i in range(1, n+1):
        fact *= i
    return fact

n = int(input("Enter number: "))
print("Factorial =", factorial(n))
```

17. Fibonacci series up to N terms

```
n = int(input("Enter N: "))
a, b = 0, 1

for _ in range(n):
    print(a, end=" ")
    a, b = b, a + b
```

18. Check Armstrong number

(Example: 153 → $1^3 + 5^3 + 3^3 = 153$)

```
num = int(input("Enter number: "))
temp = num
total = 0
```

```
while temp > 0:
    digit = temp % 10
    total += digit ** 3
    temp //= 10

if total == num:
    print("Armstrong")
else:
    print("Not Armstrong")
```

19. Merge two lists and sort them

```
a = [5, 2, 9, 1]
b = [7, 3, 6]

merged = a + b
merged.sort()

print("Merged & Sorted:", merged)
```

20. Check if two strings are anagrams

(Anagram = same characters, different order)

```
s1 = input("Enter first string: ")
s2 = input("Enter second string: ")

if sorted(s1) == sorted(s2):
    print("Anagram")
else:
    print("Not Anagram")
```

21. Find largest odd number in a list

```
lst = [12, 47, 9, 4, 27, 18]

odds = [i for i in lst if i % 2 != 0]
print("Largest odd:", max(odds))
```

22. Replace all spaces in a string with hyphens

```
s = input("Enter string: ")
print(s.replace(" ", "-"))
```

23. Maximum occurring character in a string

```
s = input("Enter string: ")

freq = {}
for ch in s:
    freq[ch] = freq.get(ch, 0) + 1

max_char = max(freq, key=freq.get)
print("Max occurring character:", max_char)
```

24A. Check if a list is sorted

```
lst = [1, 2, 3, 5, 7]

if lst == sorted(lst):
    print("List is sorted")
else:
    print("List is not sorted")
```

24B. Print all prime numbers in a given range

```
start = int(input("Start: "))
end = int(input("End: "))

for num in range(start, end + 1):
    if num > 1:
        for i in range(2, num):
            if num % i == 0:
                break
        else:
            print(num, end=" ")
```

24C. Recursive function to find sum of digits

```
def sum_digits(n):
    if n == 0:
        return 0
    return (n % 10) + sum_digits(n // 10)

num = int(input("Enter number: "))
print("Sum of digits =", sum_digits(num))
```

25. Function to count how many prime numbers are in a list

```
def is_prime(n):
    if n < 2:
        return False
    for i in range(2, n):
        if n % i == 0:
            return False
    return True

def count_primes(lst):
    count = 0
    for num in lst:
        if is_prime(num):
            count += 1
    return count

numbers = [2, 3, 4, 5, 6, 11]
print("Prime count:", count_primes(numbers))
```

26. Capitalize the first letter of each word

```
s = input("Enter sentence: ")
print(s.title())
```

27. Check if two strings have the same character frequency

(Meaning: “listen” and “silent” → same frequency)

```
s1 = input("Enter first string: ")
s2 = input("Enter second string: ")

if sorted(s1) == sorted(s2):
    print("Yes, same frequency")
else:
    print("No, frequency different")
```

28. Find all duplicate elements in a list

```
lst = [1, 2, 3, 2, 4, 5, 1, 6]

duplicates = []
for i in lst:
    if lst.count(i) > 1 and i not in duplicates:
        duplicates.append(i)

print("Duplicates:", duplicates)
```

29. Print elements that appear only once in a list

```
lst = [1, 2, 3, 2, 4, 5, 1, 6]

unique = [i for i in lst if lst.count(i) == 1]
print("Unique elements:", unique)
```

30. Print diamond-shaped star pattern

```
n = 5

# Upper part
for i in range(1, n+1):
    print(" "*(n-i) + "*"*(2*i-1))

# Lower part
for i in range(n-1, 0, -1):
    print(" "*(n-i) + "*"*(2*i-1))
```

31. Find the most frequent element in a list

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

freq = {}
for x in lst:
    freq[x] = freq.get(x, 0) + 1

most_freq = max(freq, key=freq.get)
print("Most frequent element:", most_freq)
```

32. Sum of even-indexed and odd-indexed elements

(Index starts from 0)

```
lst = [10, 20, 30, 40, 50]
```

```
even_sum = 0
odd_sum = 0

for i in range(len(lst)):
    if i % 2 == 0:
        even_sum += lst[i]
    else:
        odd_sum += lst[i]

print("Even-index sum:", even_sum)
print("Odd-index sum:", odd_sum)
```

33. Difference between largest and smallest number in a list

```
lst = [10, 2, 50, 7, 30]

difference = max(lst) - min(lst)
print("Difference:", difference)
```

Added More Questions

1. Program to find square root using `math.sqrt()`

```
import math

num = float(input("Enter a number: "))
print("Square root =", math.sqrt(num))
```

2. Question: Compute trigonometric values (sin, cos, tan) of an angle

Question form as asked:

"Write a program to compute the trigonometric values (sin, cos, tan) of an angle entered by the user using the math module."

Solution

```
import math

angle = float(input("Enter angle in degrees: "))
r = math.radians(angle)

print("sin =", math.sin(r))
print("cos =", math.cos(r))
print("tan =", math.tan(r))
```

3. Program to compute x^y using math.pow()

```
import math

x = float(input("Enter x: "))
y = float(input("Enter y: "))
```

```
print("x^y =", math.pow(x, y))
```

4. Program to get current working directory using os

```
import os

print("Current working directory:", os.getcwd())
```

5. List all files and folders using os.listdir()

```
import os

path = input("Enter directory path: ")
print(os.listdir(path))
```

6. Question: Create a folder named “Backup” using os.makedirs()

Question form as requested:

“Write a program to create a folder named ‘Backup’ inside the current directory using os.makedirs().”

Solution

```
import os

os.makedirs("Backup", exist_ok=True)
print("Backup folder created.")
```

7. Check if a path exists using os.path.exists()

```
import os

path = input("Enter path: ")

if os.path.exists(path):
    print("Path exists.")
else:
    print("Path does not exist.")
```

8. Program to rename a file using os.rename()

```
import os

old = input("Enter old file name: ")
new = input("Enter new file name: ")

os.rename(old, new)
print("File renamed.")
```

9. Copy a file using shutil.copy()

```
import shutil
```

```
src = input("Enter source file: ")
dest = input("Enter destination folder: ")

shutil.copy(src, dest)
print("File copied successfully.")
```

10. Delete a directory permanently using shutil.rmtree()

```
import shutil

folder = input("Enter folder name to delete: ")

shutil.rmtree(folder)
print("Folder deleted permanently.")
```

11. Question: Move a file using shutil.move()

Question form:

"Write a program to move a file from one folder to another using shutil.move()."

Solution

```
import shutil

src = input("Enter file path: ")
dest = input("Enter destination folder: ")

shutil.move(src, dest)
print("File moved.")
```

12. Check disk usage statistics using shutil.disk_usage()

```
import shutil

path = "/"

usage = shutil.disk_usage(path)
print("Total:", usage.total)
print("Used:", usage.used)
print("Free:", usage.free)
```

13. Program to print current system time using time.ctime()

```
import time

print("Current system time:", time.ctime())
```

14. Measure execution time using time.time()

```
import time

start = time.time()

# Code whose time you want to measure
for i in range(1000000):
```

```
    pass

end = time.time()
print("Execution time:", end - start, "seconds")
```

15. Question using time.sleep() (question form + solution)

Question (as requested):

"Write a program that uses time.sleep() to pause the program for 3 seconds before printing a message."

Solution

```
import time

print("Waiting for 3 seconds...")
time.sleep(3)
print("Done!")
```

16. Convert timestamp into readable format using time.strftime()

```
import time

timestamp = time.time()
readable = time.strftime("%Y-%m-%d %H:%M:%S", time.localtime(timestamp))

print("Readable format:", readable)
```

17. Get processor time using time.process_time()

```
import time

start = time.process_time()

for i in range(1000000):
    pass

end = time.process_time()
print("Processor time:", end - start)
```

18A. Add two integer command-line arguments

(Using `argparse`)

```
import argparse

parser = argparse.ArgumentParser()
parser.add_argument("a", type=int)
parser.add_argument("b", type=int)
args = parser.parse_args()

print("Sum =", args.a + args.b)
```

18B. Count total lines in a text file

```
filename = input("Enter file name: ")

with open(filename, "r") as f:
    lines = f.readlines()

print("Total lines:", len(lines))
```

19. Write multiple lines into a file using a loop

```
with open("data.txt", "w") as f:
    for i in range(5):
        f.write(f"Line {i+1}\n")
```

20. Question to append text at end of an existing file

Question (as required):

"Write a program to append new text at the end of an existing file using file handling."

Solution

```
with open("data.txt", "a") as f:
    f.write("This line is appended.\n")
```

21A. Read a file line-by-line using with open()

```
with open("data.txt", "r") as f:
    for line in f:
        print(line.strip())
```

21B. Question for multilevel inheritance (Grandparent → Parent → Child)

Question (as required):

"Write a program to demonstrate multilevel inheritance (Grandparent → Parent → Child)."

Solution

```
class Grandparent:
    def show_gp(self):
        print("I am Grandparent")

class Parent(Grandparent):
    def show_p(self):
        print("I am Parent")

class Child(Parent):
    def show_c(self):
        print("I am Child")

c = Child()
c.show_gp()
c.show_p()
```

```
c.show_c()
```

22. Program showing multiple inheritance

```
class A:
    def show_a(self):
        print("Class A")

class B:
    def show_b(self):
        print("Class B")

class C(A, B):
    pass

obj = C()
obj.show_a()
obj.show_b()
```

23. Question where child overrides parent's method

Question (as required):

“Write a program where a child class overrides a method of its parent class.”

Solution

```
class Parent:
    def display(self):
        print("Parent display")

class Child(Parent):
    def display(self):
        print("Child display (overridden)")

obj = Child()
obj.display()
```