**1. List Comprehension: Generating Even Numbers**

**Case Study:**

Write a Python program to generate a list of even numbers from 1 to 50 using list comprehension.

**Solution:**

even\_numbers = [num for num in range(1, 51) if num % 2 == 0]

print(even\_numbers)

**2. Dictionary Comprehension: Squaring Numbers**

**Case Study:**

Create a dictionary where keys are numbers from 1 to 10 and values are their squares.

**Solution:**

squares = {num: num\*\*2 for num in range(1, 11)}

print(squares)

**3. Regex: Extracting Emails from Text**

**Case Study:**

Write a Python program that extracts all email addresses from a given text.

**Solution:**

import re

text = "Contact us at support@example.com and sales@example.com for inquiries."

emails = re.findall(r"[a-zA-Z0-9.\_%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,}", text)

print(emails)

**4. Format Operators: Formatting a Receipt**

**Case Study:**

Format a bill receipt using the % operator.

**Solution:**

item = "Laptop"

price = 750.99

quantity = 2

receipt = "Item: %s | Quantity: %d | Price per item: $%.2f | Total: $%.2f" % (item, quantity, price, quantity \* price)

print(receipt)

**5. Hashing: Secure Password Storage**

**Case Study:**

Write a Python program to hash a password using SHA256.

**Solution:**

import hashlib

password = "securepassword123"

hashed\_password = hashlib.sha256(password.encode()).hexdigest()

print("Hashed Password:", hashed\_password)

**6. Map Function: Converting Celsius to Fahrenheit**

**Case Study:**

Convert a list of temperatures in Celsius to Fahrenheit using map.

**Solution:**

celsius = [0, 20, 30, 40]

fahrenheit = list(map(lambda x: (x \* 9/5) + 32, celsius))

print(fahrenheit)

**7. Filter Function: Filtering Prime Numbers**

**Case Study:**

Filter prime numbers from a list using filter.

**Solution:**

def is\_prime(num):

if num < 2:

return False

for i in range(2, int(num\*\*0.5) + 1):

if num % i == 0:

return False

return True

numbers = list(range(1, 50))

prime\_numbers = list(filter(is\_prime, numbers))

print(prime\_numbers)

**8. Reduce Function: Finding the Product of a List**

**Case Study:**

Find the product of all elements in a list using reduce.

**Solution:**

from functools import reduce

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

product = reduce(lambda x, y: x \* y, numbers)

print(product)

**9. OS Module: Listing Files in a Directory**

**Case Study:**

Use the os module to list all files in the current directory.

**Solution:**

import os

files = os.listdir('.')

print(files)

**10. Math Module: Calculating Factorial**

**Case Study:**

Find the factorial of a number using the math module.

**Solution:**

import math

num = 5

factorial = math.factorial(num)

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

**11. Datetime Module: Get Current Date and Time**

**Case Study:**

Display the current date and time.

**Solution:**

from datetime import datetime

current\_time = datetime.now()

print("Current Date & Time:", current\_time)

**12. Time Module: Measuring Execution Time**

**Case Study:**

Measure the execution time of a function.

**Solution:**

import time

start\_time = time.time()

sum(range(1000000)) # Some computation

end\_time = time.time()

print("Execution Time:", end\_time - start\_time, "seconds")

**13. Random Module: Generating a Random Password**

**Case Study:**

Generate a random alphanumeric password.

**Solution:**

import random

import string

password = ''.join(random.choices(string.ascii\_letters + string.digits, k=10))

print("Random Password:", password)

**14. Statistics Module: Finding the Mean**

**Case Study:**

Find the mean of a list of numbers.

**Solution:**

import statistics

data = [10, 20, 30, 40, 50]

mean\_value = statistics.mean(data)

print("Mean:", mean\_value)

**15. Sys Module: Command-Line Arguments**

**Case Study:**

Print command-line arguments.

**Solution:**

import sys

print("Command-line arguments:", sys.argv)

**16. Installing External Modules: Requests**

**Case Study:**

Use the requests module to fetch data from a website.

**Solution:**

import requests

response = requests.get("https://jsonplaceholder.typicode.com/posts/1")

print(response.json())

**17. Creating a Custom Module**

**Case Study:**

Create a custom module named math\_operations.py containing functions for addition and subtraction.

**Solution (math\_operations.py):**

def add(a, b):

return a + b

def subtract(a, b):

return a - b

**Usage in another script:**

import math\_operations

print(math\_operations.add(5, 3))

print(math\_operations.subtract(10, 4))

**18. Submodules: Using NumPy for Array Operations**

**Case Study:**

Use NumPy submodules for array manipulation.

**Solution:**

import numpy as np

arr = np.array([1, 2, 3, 4])

print("Array:", arr)

print("Sum:", np.sum(arr))

**19. Regular Expressions: Validate Phone Numbers**

**Case Study:**

Write a regex to validate phone numbers in the format (123) 456-7890.

**Solution:**

import re

phone\_number = "(123) 456-7890"

pattern = r"\(\d{3}\) \d{3}-\d{4}"

match = re.match(pattern, phone\_number)

if match:

print("Valid phone number")

else:

print("Invalid phone number")

**20. Using Multiple Modules: File Compression**

**Case Study:**

Compress and decompress a file using the gzip module.

**Solution:**

import gzip

text = b"Hello, this is a compressed file."

# Compress

with gzip.open("compressed.gz", "wb") as f:

f.write(text)

# Decompress

with gzip.open("compressed.gz", "rb") as f:

decompressed\_text = f.read()

print("Decompressed Text:", decompressed\_text.decode())