```
# 1. Advanced String Manipulation
text = "   Python is Awesome!   "       # Original string with extra spaces
cleaned = text.strip()                  # Remove leading/trailing spaces
cleaned = cleaned.lower()               # Convert to lowercase
cleaned  =  cleaned.replace("awesome",  "powerful")    #  Replace  'awesome'  with
'powerful'
print(cleaned)                          # Output the final result
"""
python is powerful!
"""
# Notes Section
# --------------------

# 2. Nested If Statements
score = 92  # A student's score

# First check if score is >= 90
if score >= 90:
    if score >= 95:  # Further check if score >= 95
        print("Excellent")
    else:
        print("Very Good")
else:
    print("Keep trying")
"""
Very Good
"""
# Notes Section
# --------------------

# 3. Looping with enumerate() and zip()
names = ["Alice", "Bob", "Charlie"]  # List of names
marks = [85, 90, 88]                 # Corresponding marks

# Loop through both lists together using zip(), and enumerate() to get index
for i, (name, mark) in enumerate(zip(names, marks), 1):
    print(f"{i}. {name}: {mark}")
"""
1. Alice: 85
2. Bob: 90
3. Charlie: 88
"""
# Notes Section
# --------------------

# 4. List Comprehension with Condition
nums = [1, 2, 3, 4, 5, 6]                # Original list
squared_evens = [x**2 for x in nums if x % 2 == 0]  # Square only even numbers
print(squared_evens)
"""
[4, 16, 36]
"""
# Notes Section
```

```
# --------------------

# 5. Dictionary Comprehension
words = ["apple", "banana", "cherry"]  # List of words
length_map = {word: len(word) for word in words}  # Map each word to its length
print(length_map)
"""
{'apple': 5, 'banana': 6, 'cherry': 6}
"""
# Notes Section
# --------------------

# 6. Function with Default and Keyword Arguments
def greet(name="Guest", lang="en"):  # Function with default values and optional language setting
    if lang == "en":
        print(f"Hello, {name}")
    elif lang == "fr":
        print(f"Bonjour, {name}")

greet("Alice", lang="fr")  # Call function with specific language
"""
Bonjour, Alice
"""
# Notes Section
# --------------------

# 7. Function Returning Multiple Values
def stats(numbers):  # Function to return min, max, and average from a list
    return min(numbers), max(numbers), sum(numbers)/len(numbers)

result = stats([3, 6, 9])  # Call the function
print(result)
"""
(3, 9, 6.0)
"""
# Notes Section
# --------------------

# 8. Nested Functions (Closure)
def outer(x):         # Outer function takes one argument
    def inner(y):     # Inner function uses value from outer scope
        return x + y
    return inner      # Returns the inner function

add5 = outer(5)       # Create a function that always adds 5
print(add5(10))
"""
15
"""
# Notes Section
# --------------------

# 9. Class with Inheritance
```

```
class Animal:
    def speak(self):
        return "Some sound"

class Dog(Animal):              # Inherit from Animal
    def speak(self):
        return "Bark"           # Override speak method

pet = Dog()                     # Create Dog object
print(pet.speak())
"""
Bark
"""
# Notes Section
# --------------------

# 10. Using Generators
def countdown(n):            # Generator function for countdown
    while n > 0:
        yield n       # Yield one value at a time
        n -= 1

for i in countdown(3):  # Iterate through generator
    print(i)
"""
3
2
1
"""
# Notes Section
# --------------------

# 11. Working with Sets
a = {1, 2, 3, 4}
b = {3, 4, 5, 6}

print("Union:", a | b)          # Union: all unique elements from both sets
print("Intersection:", a & b)   # Intersection: elements common to both
"""
Union: {1, 2, 3, 4, 5, 6}
Intersection: {3, 4}
"""
# Notes Section
# --------------------

# 12. Reading JSON Data
import json  # Import JSON module

data = '{"name": "Alice", "age": 25}'  # JSON string
parsed = json.loads(data)             # Convert string to dictionary
print(parsed["name"])
"""
Alice
"""
```

```
# Notes Section
# --------------------

# 13. Working with datetime
from datetime import datetime, timedelta

now = datetime.now()                    # Get current date/time
future = now + timedelta(days=7)        # Add 7 days
print("One week later:", future.strftime("%Y-%m-%d"))
"""
One week later: 2025-07-02  (example)
"""
# Notes Section
# --------------------

# 14. List Sorting with Custom Key
students = [("Alice", 90), ("Bob", 85), ("Eve", 92)]

# Sort by marks in descending order using lambda
students.sort(key=lambda x: x[1], reverse=True)
print(students)
"""
[('Eve', 92), ('Alice', 90), ('Bob', 85)]
"""
# Notes Section
# --------------------

# 15. Simple Recursion
def factorial(n):           # Recursive function to calculate factorial
    if n == 1:
        return 1            # Base case
    return n * factorial(n - 1)  # Recursive call

print(factorial(5))
"""
120
"""
# Notes Section
# --------------------
```