1. Write programs to understand the use of Python Identifiers, Keywords, Indentations, Comments in Python, Operators, Membership operator.

Ans.  
# Identifiers and Keywords (H1)

# Identifiers: variable, function, class names

# Keywords: reserved words like `def`, `class`, `if`, `else`, etc.

Code:-

name = "Ashish" # 'name' is an identifier

age = 21 # 'age' is also an identifier

def greet(): # 'def' is a keyword, 'greet' is an identifier

print("Hello", name)

greet()

Output :-



#Indentation

# Python uses indentation (spaces or tabs) to define code blocks  
Code :-

x = 10

if x > 5:

print("x is greater than 5") # indented block inside if

print("This line is also part of the if block")

print("This line is outside the if block") # no indentation

Output :-

A black text on a white background

AI-generated content may be incorrect.

#Comments

Code :-

"""

This is a

multi-line comment or docstring.

Used to describe what your code does.

"""

# Below is a sample program

a = 5 # assigning value 5 to variable a

b = 3

sum = a + b # adding a and b

print("Sum is:", sum)

Output :-   


#Operators

Code :-

a = 10

b = 3

# Arithmetic Operators

print("Addition:", a + b)

print("Subtraction:", a - b)

print("Multiplication:", a \* b)

print("Division:", a / b)

print("Modulus:", a % b)

print("Power:", a \*\* b)

# Comparison Operators

print("Is a equal to b?", a == b)

print("Is a not equal to b?", a != b)

print("Is a greater than b?", a > b)

Output :-

A screenshot of a computer

AI-generated content may be incorrect.

#Membership Operator

my\_list = [1, 2, 3, 4, 5]

print(3 in my\_list) # True

print(7 in my\_list) # False

print(10 not in my\_list) # True

my\_string = "hello"

print('h' in my\_string) # True

print('z' in my\_string) # False

Output :-

A black text on a white background

AI-generated content may be incorrect.

2. Write programs to understand the use of Python String, Tuple, List, Set, Dictionary, File input/output.

Ans.

#String

# String operations

Code :-

text = "Hello, Python!"

print(text.upper()) # Convert to uppercase

print(text.lower()) # Convert to lowercase

print(text[0:5]) # Slicing

print("Python" in text) # Membership check

print(text.replace("Python", "World"))

Output :-

A white background with black text

AI-generated content may be incorrect.

#Tuple

# Tuples are immutable collections

Code :-

my\_tuple = (10, 20, 30, 40)

print(my\_tuple[1]) # Accessing element

print(len(my\_tuple)) # Length of tuple

print(20 in my\_tuple) # Membership check

# my\_tuple[0] = 100 Error: Tuples are immutable

Output :-

A black and white image of a symbol

AI-generated content may be incorrect.

#List

# Lists are mutable

Code :-

my\_list = [1, 2, 3]

my\_list.append(4) # Add element

my\_list.remove(2) # Remove element

my\_list[0] = 100 # Modify element

print(my\_list)

print(len(my\_list))

Output :-



#Set

# Sets are unordered and store unique values

Code :-

my\_set = {1, 2, 3, 3, 4}

my\_set.add(5) # Add item

my\_set.discard(2) # Remove item if present

print(my\_set)

print(3 in my\_set) # Membership

Output :-



#Dictionary

# Dictionary stores key-value pairs  
Code :-

student = {

"name": "Ashish",

"age": 21,

"branch": "CSE"

}

print(student["name"]) # Access by key

student["age"] = 22 # Modify value

student["city"] = "Raipur" # Add new key-value

for key, value in student.items():

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

A white background with black text

AI-generated content may be incorrect.

3. Write programs to understand the use of Numpy’s Ndarray, Basic Operations, Indexing, Slicing, and Iterating, Conditions and Boolean Arrays.

Ans.

Pre-requisite :-

pip install numpy

Code :-

#Creating an ndarray

import numpy as np

arr = np.array([1, 2, 3, 4, 5])

print("1D Array:", arr)

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

print("2D Array:\n", matrix)

Output :-

A number on a white background

AI-generated content may be incorrect.

#Basic Operations

Code :-

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

b = np.array([4, 5, 6])

print("Addition:", a + b)

print("Subtraction:", a - b)

print("Multiplication:", a \* b)

print("Division:", b / a)

print("Square:", a \*\* 2)

print("Sum of all elements:", np.sum(a))

Output :-

A white background with black text

AI-generated content may be incorrect.

#Indexing, Slicing, and Iterating

Code :-

arr = np.array([10, 20, 30, 40, 50])

print("Element at index 2:", arr[2])

print("Sliced Array:", arr[1:4])

print("Iterating over array:")

for x in arr:

print(x)

Output :-

A white screen with black text

AI-generated content may be incorrect.

#Multidimensional Indexing and Slicing

Code :-

matrix = np.array([[1, 2, 3],

[4, 5, 6],

[7, 8, 9]])

print("Element at [1][2]:", matrix[1][2])

print("Row 2:", matrix[1])

print("Column 1:", matrix[:, 0])

print("Sub-matrix:\n", matrix[0:2, 1:3])

Output :-

A white background with black text

AI-generated content may be incorrect.

#Conditions and Boolean Arrays

Code :-

arr = np.array([5, 10, 15, 20, 25])

bool\_arr = arr > 15

print("Condition arr > 15:", bool\_arr)

filtered = arr[arr > 15]

print("Filtered values (arr > 15):", filtered)

# Multiple conditions

print("Values between 10 and 25:", arr[(arr > 10) & (arr < 25)])

Output :-

A close up of numbers

AI-generated content may be incorrect.

4. Write programs to understand the use of Numpy’s Shape Manipulation, Array Manipulation, Vectorization.

Ans.

#1. Shape Manipulation

Code :-

import numpy as np

# Original array

arr = np.array([[1, 2, 3], [4, 5, 6]])

print("Original shape:", arr.shape)

# Reshape

reshaped = arr.reshape(3, 2)

print("Reshaped (3x2):\n", reshaped)

# Flatten (1D)

flat = arr.flatten()

print("Flattened array:", flat)

# Transpose

transposed = arr.T

print("Transposed:\n", transposed)

Output :-

A white background with black text

AI-generated content may be incorrect.

#2. Array Manipulation

Code :-

import numpy as np

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

b = np.array([[5, 6]])

# Stack vertically (row-wise)

vstacked = np.vstack((a, b))

print("Vertically stacked:\n", vstacked)

# Stack horizontally (column-wise)

c = np.array([[5], [6]])

hstacked = np.hstack((a, c))

print("Horizontally stacked:\n", hstacked)

# Split array

arr = np.array([1, 2, 3, 4, 5, 6])

split = np.split(arr, 3)

print("Split array into 3 parts:", split)

Output :-

A white background with black text

AI-generated content may be incorrect.

#3. Vectorization

import numpy as np

# Without vectorization (using loop)

data = [1, 2, 3, 4]

squared = []

for x in data:

squared.append(x\*\*2)

print("Using loop:", squared)

# With NumPy vectorization

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

squared\_np = arr \*\* 2

print("Using NumPy vectorization:", squared\_np)

# Vectorized operations with functions

def f(x):

return x\*\*2 + 3\*x + 2

result = f(arr) # Automatically applied element-wise

print("Function vectorized result:", result)

A black text with black text

AI-generated content may be incorrect.

5. Write programs to understand the use of Numpy’s Structured Arrays, Reading and Writing Array Data on Files.

Ans.

#1. Structured Arrays in NumPy

#Structured arrays allow different data types for each column (like a database table).

import numpy as np

# Define structured data type

student\_dtype = np.dtype([

('name', 'U10'), # Unicode string (max 10 chars)

('age', 'i4'), # 32-bit integer

('marks', 'f4') # 32-bit float

])

# Create structured array

students = np.array([

('Alice', 20, 85.5),

('Bob', 22, 90.0),

('Charlie', 21, 78.2)

], dtype=student\_dtype)

# Access structured data

print("All student names:", students['name'])

print("Age of second student:", students[1]['age'])

print("All student records:\n", students)  
  
Output :-

A white background with black text

AI-generated content may be incorrect.

#2. Writing Array Data to File (Text File)

Code :-

import numpy as np

arr = np.array([[10, 20, 30],

[40, 50, 60]])

# Save as plain text

np.savetxt("my\_array.txt", arr, fmt='%d')

print("Array saved to my\_array.txt")  
  
Output :-



#3. Reading Array Data from File (Text File)  
Code :-

import numpy as np

# Read from text file

loaded\_arr = np.loadtxt("my\_array.txt", dtype=int)

print("Loaded array:\n", loaded\_arr)

Output :-

A close up of numbers

AI-generated content may be incorrect.

#4. Save & Load as Binary (npy format)

Code :-

import numpy as np

arr = np.array([[1, 2, 3], [4, 5, 6]])

# Save binary file

np.save('my\_binary\_array.npy', arr)

# Load binary file

loaded = np.load('my\_binary\_array.npy')

print("Loaded from .npy file:\n", loaded)  
  
Output ;-

A black and white text

AI-generated content may be incorrect.