**📘 List Assignments – Demonstration & Explanations**

**✅ 1. Create and Display a List**

lst = [20, 12, 4, 23, 21, 32, 11]

print(lst)

**Creates** a list of integers and prints it.

**✅ 2. Print First, Middle, and Last Elements**

print(f"First element: {lst[0]}")

lstlen = len(lst) // 2

print(f"Middle element: {lst[lstlen]}")

print(f"Last element: {lst[-1]}")

Accesses and prints the **first**, **middle**, and **last** elements using indexing.

**✅ 3. Slicing Elements**

print(lst[:5]) # First 5 elements

print(lst[-5:]) # Last 5 elements

print(lst[5:16]) # Elements from index 5 to 15

Demonstrates **list slicing** to extract specific segments.

**✅ 4. Squares of First 10 Positive Integers (List Comprehension)**

squarlist = [x\*\*2 for x in range(1, 11)]

print(squarlist)

Generates a list of squares using **list comprehension**.

**✅ 5. Filter Even Numbers from a List**

evennumberlist = [x for x in squarlist if x % 2 == 0]

print(evennumberlist)

Filters out **even numbers** from the list of squares.

**✅ 6. Random Numbers List: Sorting & Removing Duplicates**

import random

lst = [x for x in range(10) if x != random.randint(0, 10)]

lst.sort()

print(lst)

lst.reverse()

print(lst)

Creates a list using random exclusion, **sorts** it in ascending and descending order.

**✅ 7. Nested List (3x3 Matrix) and Access**

list\_nested = [[x, y] for x in range(10) for y in range(10)]

print(list\_nested)

# Access example for matrix:

matrix = [[1,2,3], [4,5,6], [7,8,9]]

print(matrix[1][2]) # Second row, third column

Demonstrates creation of **nested lists (matrices)** and element access.

**✅ 8. List Comprehension with Function Call**

word = ["hello", "world", "", "list", "comprehension"]

lengths = [len(w) for w in word]

print(sorted(set(lengths)))

Computes the length of each word and removes duplicates.

**✅ 9. List of Dictionaries Sorted by Key**

dic\_student = [

{"name": "a", "score": 200},

{"name": "b", "score": 100},

{"name": "c", "score": 300}

]

sorted\_student = sorted(dic\_student, key=lambda student: student["score"])

print(sorted\_student)

Sorts a list of dictionaries based on the 'score' key.

**✅ 10. Transpose a 3×3 Matrix**

original\_matrix = [

[1, 2, 3],

[4, 5, 6],

[7, 8, 9]

]

transposed\_matrix = [[original\_matrix[y][x] for y in range(3)] for x in range(3)]

print("Transposed Matrix:")

for row in transposed\_matrix:

print(row)

Uses **nested list comprehension** to transpose a matrix.

**✅ 11. Reverse and Rotate a List**

lst = [x for x in range(1, 8)]

print("Original:", lst)

rotateby = -1

right\_rotated = lst[-rotateby:] + lst[:-rotateby]

left\_rotated = lst[rotateby:] + lst[:rotateby]

print("Right Rotated:", right\_rotated)

print("Left Rotated:", left\_rotated)

Demonstrates reversing and rotating a list using **slicing**.

**Interactive Version:**

def rotate\_list(lst, n):

return lst[n:] + lst[:n]

lst = [x for x in range(1, 8)]

n = int(input("Enter number to rotate: "))

print("Rotated List:", rotate\_list(lst, n))

**✅ 12. Flatten a Nested List**

nested\_list = [

[1, 2, 3],

[4, 5, 6],

[7, 8, 9]

]

flat\_list = [item for sublist in nested\_list for item in sublist]

print("Flattened List:", flat\_list)

Flattens a nested 2D list into a single flat list.

**✅ 13. Remove Items and Insert at Index**

list\_no = [item for item in range(1, 10)]

del list\_no[2]

del list\_no[3]

del list\_no[4]

list\_no.insert(5, 99)

print(list\_no)

**Removes** elements at specific indices and **inserts** a new one.

**✅ 14. List Intersection (Common Elements)**

list1 = [1, 2, 3, 4, 5]

list2 = [3, 4, 5, 6, 7]

intersected = [x for x in list1 if x in list2]

print("Intersection:", intersected)

Finds common elements between two lists using comprehension.

**📝 Final Notes:**

* Code demonstrates **core list operations**: creation, indexing, slicing, sorting, comprehension, nesting, transposing, and more.
* Applies both **basic** and **advanced** techniques for manipulating lists.