**Python Lab Set-6 feb,6**

**Name: Gaurang Bhadane**

**Prn no:-1272240276**

**Date: 6-Feb-2025**

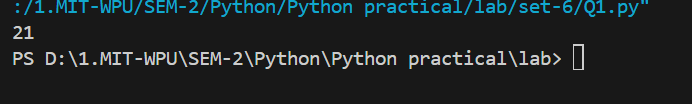
1. **Create a tuple with a list of tuples. Each  
   tuple should contain two numbers. Calculate the sum of all the numbers in the  
   inner tuples and return the result.**

Code:-

tuple\_list = ((1, 2), (3, 4), (5, 6))

sum\_all\_numbers = sum([sum(t) for t in tuple\_list])

sum\_all\_numbers



1. **Create a list that contains multiple  
   tuples. Each tuple should contain two elements. Write a function that merges  
   the tuples into a single list by combining elements from each tuple.**

**Code:-**

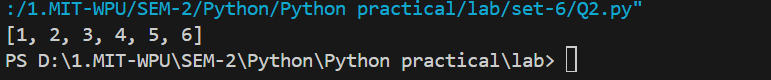
def merge\_tuples(tuples):

    return [item for t in tuples for item in t]

tuples = [(1, 2), (3, 4), (5, 6)]

merged = merge\_tuples(tuples)

print(merged)



1. **Write a function that accepts a tuple  
   containing a mix of lists and tuples. The function should return a new tuple  
   where all the inner elements (lists and tuples) are flattened into a single  
   list and the tuple itself is replaced by this flattened list.**

Code:-

def flatten\_tuple(t):

    return [item for sub in t for item in (sub if isinstance(sub, (tuple, list)) else [sub])]

nested\_tuple = ((1, 2), [3, 4], (5, 6))

flattened = flatten\_tuple(nested\_tuple)

print(flattened)  # Output: [1, 2, 3, 4, 5, 6]



1. **Create a list of tuples where each  
   tuple contains a string and an integer. Write a function that sorts the list of  
   tuples first by the string and then by the integer.**

Code:-

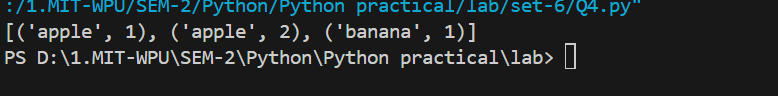
def sort\_tuples(tuples):

    return sorted(tuples, key=lambda x: (x[0], x[1]))

tuples = [("apple", 2), ("banana", 1), ("apple", 1)]

sorted\_tuples = sort\_tuples(tuples)

print(sorted\_tuples)  # Output: [('apple', 1), ('apple', 2), ('banana', 1)]



**5. Create a nested tuple with a list  
inside it. Use list operations to add and remove elements from the list inside  
the tuple, while ensuring the tuple itself remains immutable.**

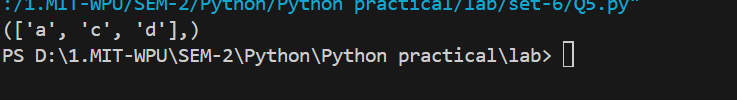
Code

nested\_tuple = (["a", "b", "c"],)

nested\_tuple[0].append("d")

nested\_tuple[0].remove("b")

print(nested\_tuple)  # Output: (['a', 'c', 'd'],)



**6. You are given a , where the keys are department names, and the values are dictionaries containing as keys and their list of courses as values.**

* **Allow dynamic input for multiple departments and professors.**
* **Retrieve the list of courses taught by a given professor.**
* **Add a new professor to an existing department.**

Code:

departments = {}

def add\_department(name):

    departments[name] = {}

def add\_professor(department, professor, courses):

    departments[department][professor] = courses

def get\_courses(professor):

    for dept in departments.values():

        if professor in dept:

            return dept[professor]

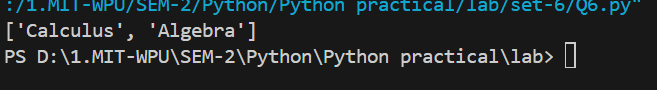
add\_department("Math")

add\_professor("Math", "Dr. Smith", ["Calculus", "Algebra"])

add\_professor("Math", "Dr. Lee", ["Geometry"])

courses = get\_courses("Dr. Smith")

print(courses)  # Output: ['Calculus', 'Algebra']



**7. Write a program to store employee details in a dictionary where:**

* **The keys are employee IDs.**
* **The values are Name, Address, Position, Joining Date**
* **Allow users to retrieve employee details based on the ID.**

Code:-

employees = {}

def add\_employee(emp\_id, name, position):

    employees[emp\_id] = {"name": name, "position": position}

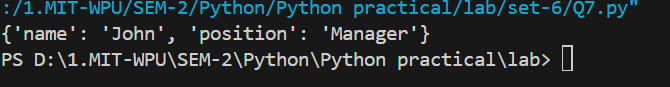
def get\_employee(emp\_id):

    return employees.get(emp\_id)

add\_employee(1, "John", "Manager")

emp\_details = get\_employee(1)

print(emp\_details)  # Output: {'name': 'John', 'position': 'Manager'}



**8. Create a dictionary where the keys are student names and the values are lists of their subject grades.**

* **Take dynamic input for multiple students and their grades.**
* **Retrieve and print the grades of a given student**

Code:-

students = {}

def add\_student(name, grades):

    students[name] = grades

def get\_grades(name):

    return students.get(name)

add\_student("Alice", [90, 80, 85])

grades = get\_grades("Alice")

print(grades)  # Output: [90, 80, 85]

