***Name – Aryan Sharma***

***BT21GCS161***

***ASSIGNMENT - 09***

***Code-***

from tabulate import tabulate

def bankers\_algorithm(processes, allocation, maximum, available):

    n = len(processes)

    m = len(available)

    # Initialize data structures

    remaining = [[maximum[p][q] - allocation[p][q] for q in range(m)] for p in range(n)]

    SS = [] # safe\_sequence

    W = available[:] # work

    finish = [False] \* n

    # Find a safe sequence

    while True:

        found = False

        for p in range(n):

            if not finish[p] and all(remaining[p][q] <= W[q] for q in range(m)):

                W = [W[q] + allocation[p][q] for q in range(m)]

                finish[p] = True

                SS.append(processes[p])

                found = True

        if not found:

            break

    # Check if the safe sequence covers all processes

    if all(finish):

        table\_headers = ["Process", "Allocation", "Max Need", "Available", "Remaining"]

        table\_data = []

        for p in range(n):

            row = [processes[p], allocation[p], maximum[p], available, remaining[p]]

            table\_data.append(row)

        print(tabulate(table\_data, headers=table\_headers, tablefmt="fancy\_grid"))

        print("\nSafe Sequence:", SS)

    else:

        print("The system is in an unsafe state.")

# Example usage

processes = ['1', '2', '3', '4', '5']

allocation = [[0, 0, 1, 2], [1, 0, 0, 0], [1, 3, 5, 4], [0, 6, 3, 2], [0, 0, 1, 4]]

maximum = [[0, 0, 1, 2], [1, 7, 5, 0], [2, 3, 5, 6], [0, 6, 5, 2], [0, 6, 5, 6]]

available = [1, 5, 2, 0]

bankers\_algorithm(processes, allocation, maximum, available)

***Output –***

