Name: Mudit Mohit Assignment 9

Enrolment Number: BT21GCS057

Section: B2

Q.) WAP to simulate banker’s algorithm

Ans.

from typing import List, Tuple

def is\_safe(processes: List[int], available: List[int], allocated: List[List[int]], need: List[List[int]]) -> bool:

work = available[:]

finish = [False] \* len(processes)

while True:

done = True

for i in range(len(processes)):

if not finish[i] and all(need[i][j] <= work[j] for j in range(len(available))):

work = [work[j] + allocated[i][j] for j in range(len(available))]

finish[i] = True

done = False

if done:

break

return all(finish)

def bankers\_algorithm(processes: List[int], available: List[int], allocated: List[List[int]], need: List[List[int]]) -> Tuple[bool, List[int]]:

if not is\_safe(processes, available, allocated, need):

return False, []

work = available[:]

finish = [False] \* len(processes)

sequence = []

while True:

done = True

for i in range(len(processes)):

if not finish[i] and all(need[i][j] <= work[j] for j in range(len(available))):

work = [work[j] + allocated[i][j] for j in range(len(available))]

finish[i] = True

sequence.append(processes[i])

done = False

if done:

break

return True, sequence

# Example inputs

processes = [0, 1, 2, 3, 4]

available = [3, 3, 2]

allocated = [[0, 1, 0], [2, 0, 0], [3, 0, 2], [2, 1, 1], [0, 0, 2]]

need = [[7, 4, 3], [0, 2, 0], [6, 0, 0], [0, 1, 1], [4, 3, 1]]

# Run the Banker's algorithm

is\_safe, sequence = bankers\_algorithm(processes, available, allocated, need)

# Print the output

if is\_safe:

print("The state is safe.")

print("The processes should be executed in the following sequence:", sequence)

else:

print("The state is unsafe.")

Output:

