

Lab Experiment 02

Sushil Shetty

22BAI10297

Banker's Algorithm

Code :

```
def is_safe(processes, allocation, max_resources, available):

    need = [[max_resources[i][j] - allocation[i][j] for j in range(len(available))]
for i in range(processes)]
    finish = [False] * processes
    work = available.copy()
    safe_seq = []

    while any(not f for f in finish):
        found = False
        for i in range(processes):
            if (not finish[i]) and all(need[i][j] <= work[j] for j in
range(len(available))):
                for j in range(len(available)):
                    work[j] += allocation[i][j]
                finish[i] = True
                safe_seq.append(i)
                found = True
        if not found:
            return None

    return safe_seq

# Example usage
processes = 5
resources = 3
allocation = [
    [0, 1, 0],
    [2, 0, 0],
    [3, 0, 2],
    [2, 1, 1],
    [0, 0, 2]
```

```

]
max_resources = [
    [7, 5, 3],
    [3, 2, 2],
    [9, 0, 2],
    [2, 2, 2],
    [4, 3, 2]
]
available = [3, 3, 2]

safe_sequence = is_safe(processes, allocation, max_resources, available)

if safe_sequence:
    print("Following is the SAFE Sequence:")
    for process in safe_sequence:
        print(" P", process, end=" -> ")
    print("P", safe_sequence[-1])
else:
    print("The following system is not safe")

```

Output :

```

● PS C:\Users\skaro> & C:/Users/skaro/AppData/Local/Microsoft/WindowsApps/python3.11.exe f:/Coding/Codes/Python/Bankers.py
Following is the SAFE Sequence:
  P 1 -> P 3 -> P 4 -> P 0 -> P 2 -> P 2
○ PS C:\Users\skaro>

```