# CSE :- 316

**Final Report**

**TOPIC:-**

Generate a solution to find whether the system is in safe state or not?

**By :-**

* + **Name :-** Divya Raushan
  + **Reg.No:-**11812258
  + **Section :-** K18GA
  + **Email:-**hs68217@gmail.com
  + **GitHub Link:**-

**Submitted To:-** Navneet Kaur Mam

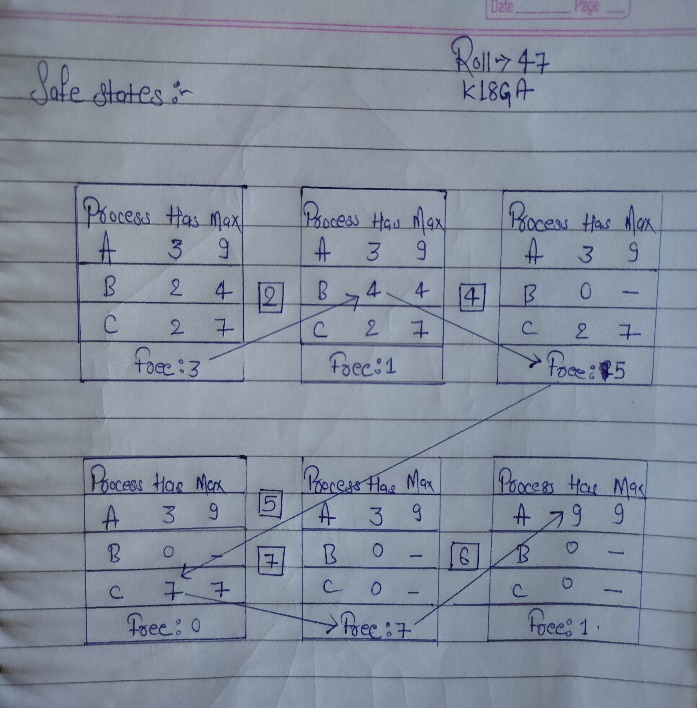
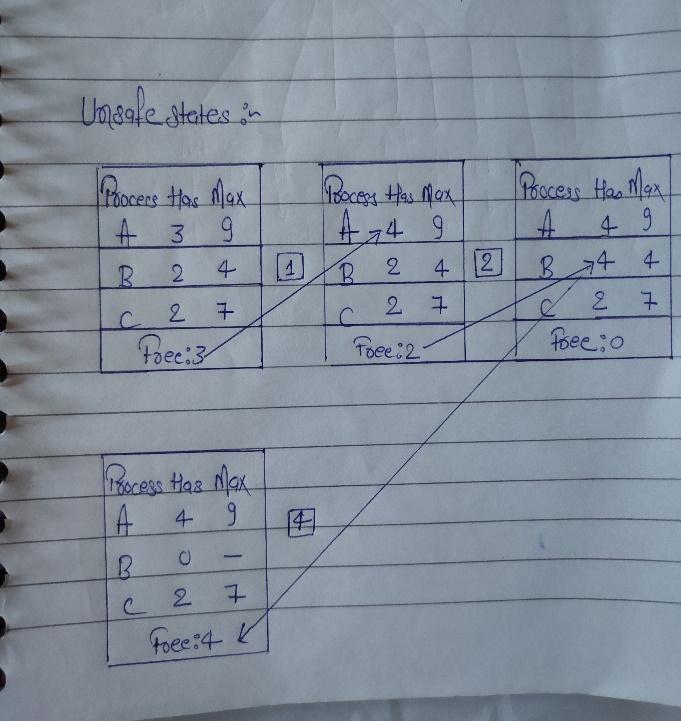
**Introduction :-**

**Safe and Unsafe state**

A state is safe if the system can allocate all resources requested by all processes ( up to their stated maximums ) without entering a deadlock state.

More formally, a state is safe if there exists a safe sequence of processes { P0, P1, P2, ..., PN } such that all of the resource requests for Pi can be granted using the resources currently allocated to Pi and all processes Pj where j is less than i. ( I.e. if all the processes prior to Pi finish and free up their resources, then Pi will be able to finish also, using the resources that they have freed up. )

If a safe sequence does not exist, then the system is in an unsafe state, which MAY lead to deadlock. ( All safe states are deadlock free, but not all unsafe states lead to deadlocks.)The banker’s algorithm is a resource allocation and deadlock avoidance algorithm that tests for safety by simulating the allocation for predetermined maximum possible amounts of all resources, then makes an “s-state” check to test for possible activities, before deciding whether allocation should be allowed to continue.

****

# Banker’s Algorithm in Operating System

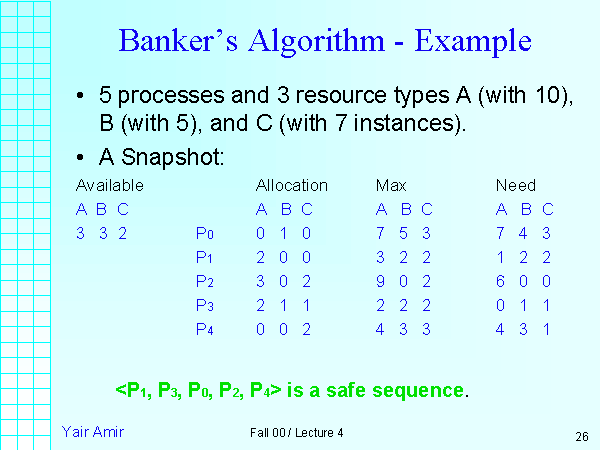
The algorithm that test whether the system is in safe state or not by simulating the allocation for predetermined maximum possible amounts of all resources is called **Banker’s Algorithm**

The banker’s algorithm is a resource allocation and deadlock avoidance algorithm that tests for safety by simulating the allocation for predetermined maximum possible amounts of all resources, then makes an “s-state” check to test for possible activities, before deciding whether allocation should be allowed to continue.

**Why Banker’s algorithm is named so?**

Banker’s algorithm is named so because it is used in banking system to check whether loan can be sanctioned to a person or not. Suppose there are n number of account holders in a bank and the total sum of their money is S. If a person applies for a loan then the bank first subtracts the loan amount from the total money that bank has and if the remaining amount is greater than S then only the loan is sanctioned. It is done because if all the account holders comes to withdraw their money then the bank can easily do it.

In other words, the bank would never allocate its money in such a way that it can no longer satisfy the needs of all its customers**. The bank would try to be in safe state always**.



**Following Data structures are used to implement the Banker’s Algorithm** **:-**

Let **‘n’**be the number of processes in the system and **‘m’**be the number of resources types.

**Available :**

* It is a 1-d array of size **‘m’** indicating the number of available resources of each type.
* Available[ j ] = k means there are **‘k’** instances of resource type **Rj**

**Max :**

* It is a 2-d array of size ‘**n\*m’**that defines the maximum demand of each process in a system.
* Max[ i, j ] = k means process **Pi** may request at most **‘k’** instances of resource type **Rj.**

**Allocation :**

* It is a 2-d array of size**‘n\*m’**that defines the number of resources of each type currently allocated to each process.
* Allocation[ i, j ] = k means process **Pi** is currently allocated **‘k’** instances of resource type **Rj**

**Need :**

* It is a 2-d array of size **‘n\*m’** that indicates the remaining resource need of each process.
* Need [ i,   j ] = k means process **Pi** currently need **‘k’** instances of resource type **Rj**

for its execution.

* Need [ i,   j ] = Max [ i,   j ] – Allocation [ i,   j ]

Allocationi specifies the resources currently allocated to process Pi and Needi specifies the additional resources that process Pi may still request to complete its task.

Banker’s algorithm consists of Safety algorithm and Resource request algorithm/

**Safety Algorithm :-**

The algorithm for finding out whether or not a system is in a safe state can be described as follows:

1) Let Work and Finish be vectors of length ‘m’ and ‘n’ respectively.  
Initialize: Work = Available  
Finish[i] = false; for i=1, 2, 3, 4….n

This means, initially, no process has finished and the number of available resources is represented by the **Available** array.

2) Find an i such that both  
a) Finish[i] = false  
b) Needi <= Work  
if no such i exists goto step (4)

It means, we need to find an unfinished process whose need can be satisfied by the available resources. If no such process exists, just go to step 4.

3) Work = Work + Allocation[i]  
Finish[i] = true go to step (2)

When an unfinished process is found, then the resources are allocated and the process is marked finished. And then, the loop is repeated to check the same for all other processes.

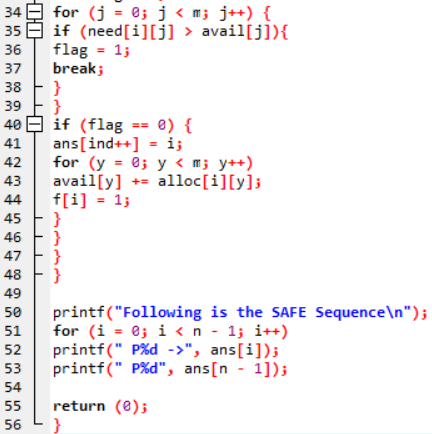
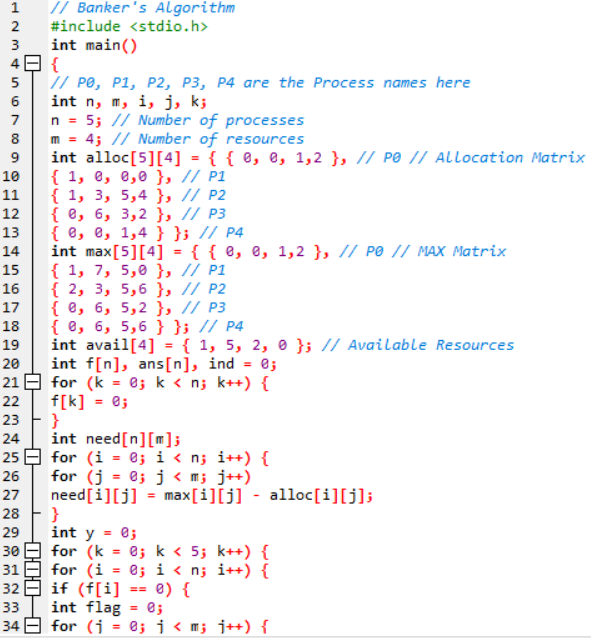
4) if Finish [i] = true for all i  
then the system is in a safe state .

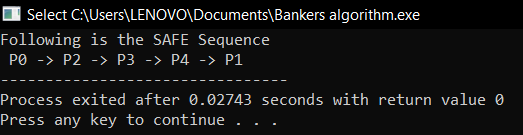
That means if all processes are finished, then the system is in safe state.

**Time Complexity :-**

The time complexity of banker’s algorithm is m\*(n\*n) where n is the number of active processes and m is the number of resources.

**Code for Banker’s Algorithm :-**



**Output :-**

The below part introduces (n\*m) time complexity

for I = 1 to N do // \*n times

if ((not FINISH[i]) and

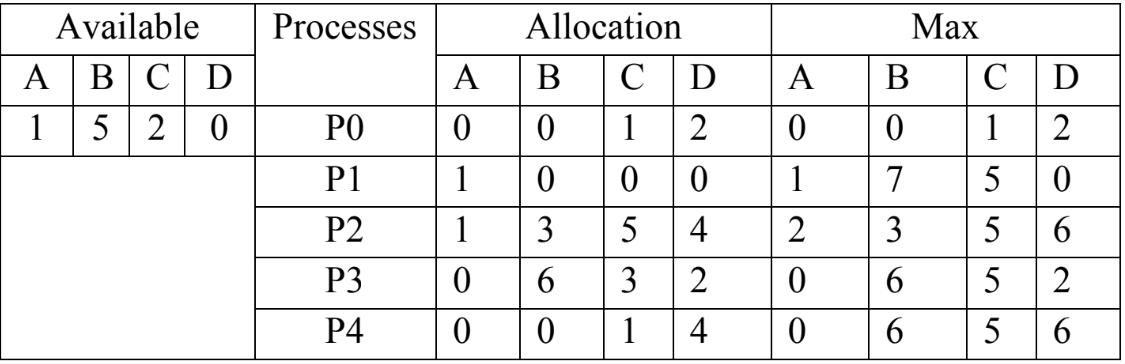
NEEDi <= WORK) then // \*m times, if it's an array search

but it is also nested in a repeat loop. If that loop can run, in worst case, n times, then the procedure has O(n\*n\*m) time complexity.

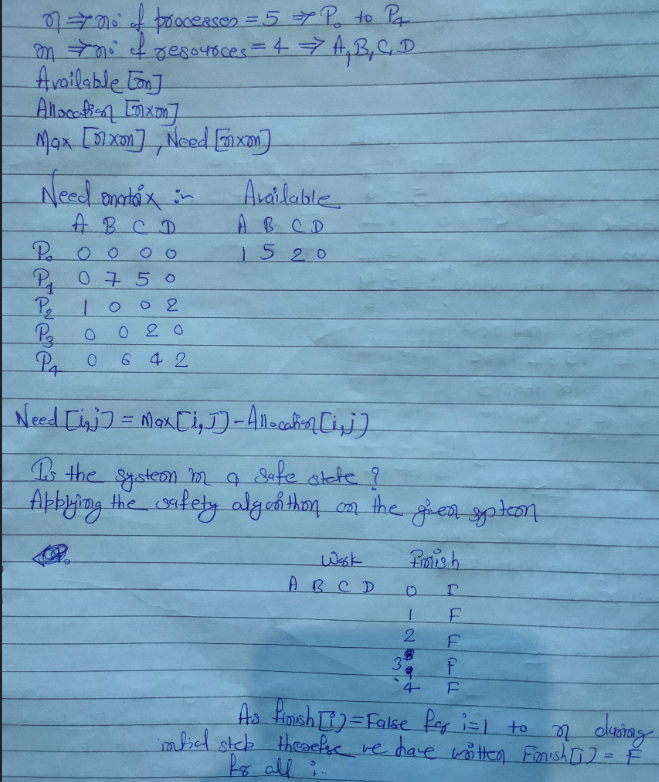
WORK = WORK + ALLOCATION\_i; // also O(m) operation, vectors addition

So, the code that executes in the for loop has O(m+m) time complexity.  
Of course, O(m+m) = O(m) and the final complexity is O(n\*n\*m).

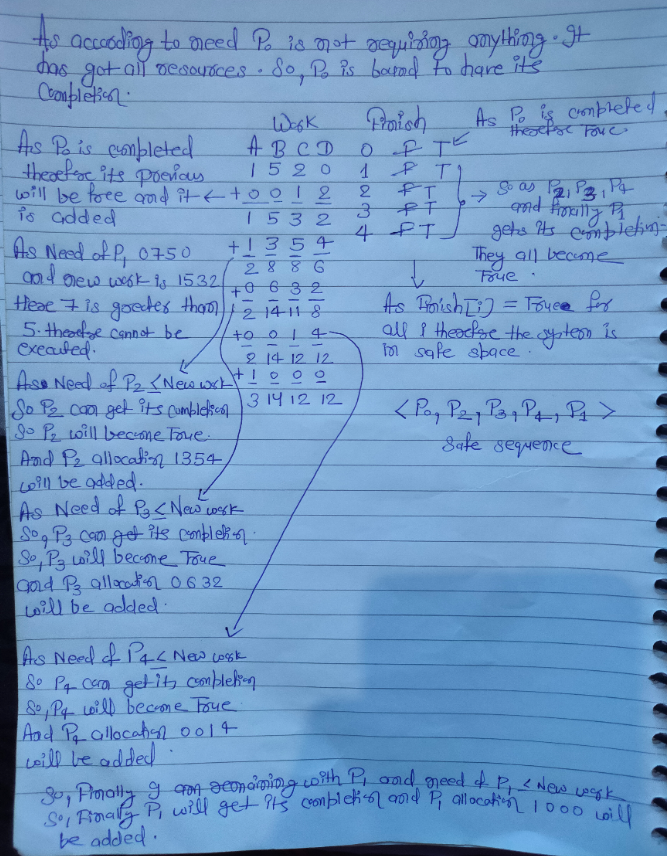
**Solution of assigned problem and Its Explanation :-**



**Need matrix :**



**Applying the safety algorithm on the given system :**



Therefor,the system is in safe state and the safe sequence is < P0,P2,P3,P4,P1 >.