EXPERIMENT - 8

Operating Systems Lab

AIM

Write a program to implement Banker's algorithm for deadlock avoidance.

Syeda Reeha Quasar
14114802719
6C7

EXPERIMENT – 8

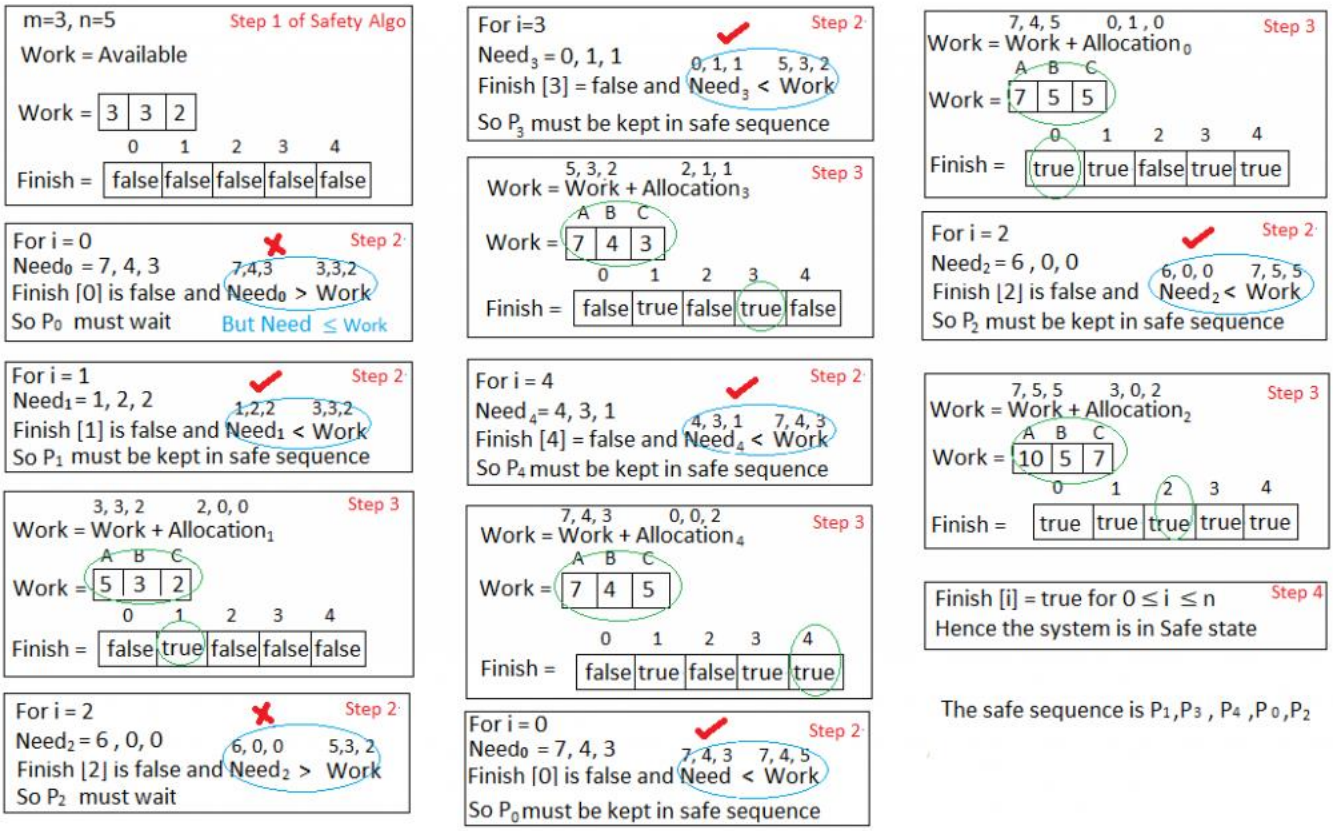
Aim:

Write a program to implement Banker's algorithm for deadlock avoidance.

Theory:

It is a banker algorithm used to **avoid deadlock** and **allocate resources** safely to each process in the computer system. The '**S-State**' examines all possible tests or activities before deciding whether the allocation should be allowed to each process. It also helps the operating system to successfully share the resources between all the processes. The banker's algorithm is named because it checks whether a person should be sanctioned a loan amount or not to help the bank system safely simulate the allocation of resources.

Example:



Source Code:

```
alloc=( 0 1 0 2 0 0 3 0 2 2 1 1 0 0 2 )
max=( 7 5 3 3 2 2 9 0 2 2 2 2 4 3 3 )
avail=(3 3 2)
n=5
m=3
declare -a f
declare -a ans
declare -a need
ind=0
```

```
i=0
while [ $i -lt $n ]
do
    f[$i]=0
    i=$((i+1))
done
a=0
while [ $a -lt $n ]
do
    b=0
    while [ $b -lt $m ]
    do
        need[ $(( a*m+b )) ]=$((max[ $(( a*m+b )) ]-alloc[ $(( a*m+b )) ]))
        b=$((b+1))
    done
    a=$((a+1))
done
p=0
while [ $p -lt 5 ]
do
    q=0
    while [ $q -lt $n ]
    do
        if [ ${f[$q]} -eq 0 ]
        then
            flag=0
            r=0
            while [ $r -lt $m ]
            do
                if [ ${need[ $(( q*m+r )) ]} -gt ${avail[$r]} ]
                then
                    flag=1
                    break
                fi
                r=$((r+1))
            done
            if [ $flag -eq 0 ]
            then
                ans[$ind]=$q
                ind=$((ind+1))
                y=0
                while [ $y -lt $m ]
                do
                    avail[$y]=$(( avail[ $y ] + alloc[ $(( q*m+y )) ] ))
                    y=$((y+1))
                done
                f[$q]=1
            fi
        fi
        q=$((q+1))
    done
```

```
    p=$((p+1))
done
flag2=1
x=0
while [ $x -lt $n ]
do
    if [ ${f[$x]} -eq 0 ]
    then
        flag2=0
        echo "The following systen is not safe"
        break
    fi
    x=$((x+1))
done
if [ $flag2 -eq 1 ]
then
    echo "Following is the SAFE sequence"
    echo "P"${ans[0]} "--> P"${ans[1]} "--> P"${ans[2]} "--> P"${ans[3]} "-->
P"${ans[4]}
fi
```

Output: