Sri Sivasubramaniya Nadar College of Engineering

(An Autonomous Institution, Affiliated to Anna University, Chennai)

Department of Computer Science and Engineering

UCS1411 – Operating Systems Laboratory

Lab Exercise 7: Implementation of Banker's Algorithm (Deadlock Avoidance) and Deadlock Detection

Aim:

Assignment 1: Develop a C program to implement Banker's algorithm for deadlock avoidance with multiple instances of resource types

Assignment 2: Develop a C program to implement algorithm for deadlock detection with multiple instances of resource types and display the processes involved in deadlock

Deadlock Avoidance Algorithm:

```
1. Read the following
        a. Number of processes.
        b. Number of resources and number of instances of each resource available.
        c. Maximum requirement of each process,
        d. Allocated instances of resources
2. Determine the need of each process
3. Display menu as
        1) Check system state
        2) Resource Request.
4. If choice is 1 do the following
        4.1 Repeat the following till all processes are done.
                a. Check if need of process i is less than or equal to available instances
                        i. If yes proceed with step 4.1.b
                        ii. Otherwise wait till available
                b. Update the available vector by adding the resources released by Pi. Add
                Process Pi to safe sequence.
        4.2. If any of the process is not able to finish, then display as "Unsafe State". Else display the safe sequence.
5. If choice is 2 do the following
        5.1 Get the process id which requests the resource and the request vector
        5.2 If request <= need of process and request <= available then
                5.3 Pretend to allocate by updating the available, need and allocated vectors
                5.4 Run safety algorithm.
                5.5 If safe, grant the resources by updating the system state. Else, display
                "Resource request by $P_i$ not granted" and revert back to original state.
```

SAMPLE INPUT & OUTPUT:

Banker's Algorithm

```
1. Read Data
2. Print Data
3. Check system state
4. Resource request
5.Exit

Enter the option :1
Number of processes: 5 P0, P1, P2, P3, P4
Number of resources: 3 A B C
Number of Available instances of A: 3
Number of Available instances of B: 3
Number of Available instances of C: 2
Maximum requirement for P0: 7 5 3
Maximum requirement for P1: 3 2 2
Maximum requirement for P2: 9 0 2
Maximum requirement for P3: 2 2 2
Maximum requirement for P4: 4 3 3
Allocated instances to P0: 0 1 0
Allocated instances to P1: 2 0 0
Allocated instances to P2: 3 0 2
Allocated instances to P3: 2 1 1
Allocated instances to P4: 0 0 2

Enter the option: 2

        Alloc           Max             Need             Avail
        A B C           A B C           A B C           A B C
P0      0 1 0           7 5 3           * * *           3 3 2
P1      2 0 0           3 2 2           * * *
P2      3 0 2           9 0 2           * * *
P3      2 1 1           2 2 2           * * *
P4      0 0 2           4 3 3           * * *

Enter the option: 3
Display the Safety Sequence:
* * * * *

Enter the option: 4

Enter the process id of process requesting for new resources: P1

Enter the request vector for P1: 4 3 3

Safe sequence is *****. Resource request by P1 can be granted
```