Page No.
Date: / /

Assignment C2

Title : Banker's Algorithm

Date of Completion :

Problem Statement :
Write a Java Program to implement Banker's Algorithm

Objectives :
To understand bankers algorithm
To implement banker's algorithm
To use OOP concepts for the same.

Outcomes : I will be able to :
Understand Concept of Deadlock
Understand Strategies to deal with deadlock.
Implement Banker's Algorithm using concepts of OOP.

Software & Hardware Requirements :
Fedora 64 bit OS
Working PC
Eclipse IDE for Java
i5 processor
4 GB RAM
500 GB HDD.

Page No.
Date: / /

Theory:

Deadlock:
In multiprogramming environment, several processes may compete for a fixed number of resources. A process requests a resource & if the resource is not available at that time, it enters wait state. It may happen that it will never gain access to a resource, if that resource is being held by other waiting processes.
A set of processes is deadlocked if each process is waiting for a resource that is held by a deadlocked process.

Conditions for Deadlock:
1. Mutual Exclusion
2. Hold & Wait
3. No preemption
4. Circular Wait Condition.

Strategies to deal with deadlock:
1. Ignore the problem altogether.
2. Detection & Recovery
3. Dynamic Avoidance by Careful Resource allocation
4. Prevention by negating above 3 conditions.

Ignore
Detec
Detecti
Method
Dead
Dead
deadl
Approa
1.
2.

Page No.
Date: / /

ore the problem (Ostrich Algorithm)

Detection & Recovery

Detection - A deadlock detection algorithm tries to detect the presence of a cycle in a resource graph.
The algorithm terminates if
1. Cycle is found
2. Cycle doesn't exist

Method - Methods to recover from deadlock are:
1. Preemption
2. Rollback
3. Killing Processes

Deadlock Avoidance :
Deadlock Avoidance approach ensures that deadlock does not arise in a system.

Approaches include :
1. Do not start a process if its demands might lead to a deadlock.
2. Do not grant incremental resources request to a process if this allocation might lead to a deadlock.
Example - Banker's Algorithm.

Page No.
Date: / /

Banker's Algorithm :

Banker's Algorithm is the algorithm of resource allocation denial.

Tables used by Banker's Algorithm :

1. Resources in Existence (E)
2. Resources Available (A)
3. Maximum Claim Matrix (C)
4. Current Allocation Matrix (R)

Banker's Algorithm states that a request should be granted iff granting it leads to a safe state.

State of a System - is the current allocation of resources to process.

Safe State - is a state in which there is a way to ~~st~~ satisfy all the currently pending requests by running the process in some order.

Deadlock Avoidance Steps :-

Step - 1. When a process makes a request for a set of resources, allocation matrix is updated assuming that request is granted.

Step - 2 If System is in safe state then
2.1. grant request for resources
else
2.1. reverse current allocation
2.2. block the process until safe state.

Page No.
Date: / /

Databases for Bankers Algorithm

Maximum Claim Matrix	int cm[10][10]
Initial Allocation Matrix	int am[10][10]
Resource vector for resources in existence	int rv[10]
Vector for available resources after allocation	int av[10]
Vector to read the request for grant of resource for any process	int vector[10]
Number of requesting process	int pno
Number of resources	int nr
Number of processes	int np.

Test Cases

Input	Expected O/P	Actual O/P	Result
Allocation Matrix A B C 1 0 0 6 1 2 2 1 1 0 0 2 Claim Matrix A B C 3 2 2 6 1 3 3 1 4 4 2 2 Resource Vector 9 3 6	No of Safe sequences = 6 P2→P1→P3→P4 P2→P1→P4→P3 P2→P3→P1→P4 P2→P3→P4→P1 P2→P4→P1→P3 P2→P4→P3→P1	No of safe sequences = 6 P2→P1→P3→P4 P2→R1→P4→P3 P2→P3→P1→P4 P2→P3→P4→P1 P2→P4→P1→P3 P2→P4→P3→P1	Success Success

Page No.
Date: / /

Conclusion :
Hence Banker's Algorithm has been successfully implemented using concepts of OOP.