

**Vision of the Department**

To be a well-known centre for pursuing computer education through innovative pedagogy, value-based education and industry collaboration.

**Mission of the Department**

To establish learning ambience for ushering in computer engineering professionals in core and multidisciplinary area by developing Problem-solving skills through emerging technologies.

**Session 2025-2026**

<b>Vision:</b> Dream of Where you want	<b>Mission:</b> Means to achieve vision
--	---

**Program Educational Objectives of the program (PEO):** (broad statements that describe the professional and career accomplishments)

PEO1	<b>Preparation</b>	<b>P: Preparation</b>	<b>Pep-CL abbreviation pronounce as Pep-si-LL easy to recall</b>
PEO2	<b>Core Competence</b>	<b>E: Environment (Learning Environment)</b>	
PEO3	<b>Breadth</b>		
PEO4	<b>Professionalism</b>	<b>P: Professionalism</b>	
PEO5	<b>Learning</b>	<b>C: Core Competence</b>	
	<b>Environment</b>	<b>L: Breadth (Learning in diverse areas)</b>	

**Program Outcomes (PO):** (statements that describe what a student should be able to do and know by

the end of a program) **Keywords of POs:**

Engineering knowledge, Problem analysis, Design/development of solutions, Conduct Investigations of Complex Problems, Engineering Tool Usage, The Engineer and The World, Ethics, Individual and Collaborative Team work, Communication, Project Management and Finance, Life-Long Learning

**PSO Keywords:** Cutting edge technologies, Research

"I am an engineer, and I know how to apply engineering knowledge to investigate, analyse and design solutions to complex problems using tools for entire world following all ethics in a collaborative way with proper management skills throughout my life." *to contribute to the development of cutting-edge technologies and Research.*

---

**Integrity:** I will adhere to the Laboratory Code of Conduct and ethics in its entirety.

**Name and Signature of Student and Date**

(Signature and Date in Handwritten)



## Department of Computer Technology B.Tech in Computer Science and Engineering (IOT)

## Vision of the Department

To be a well-known centre for pursuing computer education through innovative pedagogy, value-based education and industry collaboration.

## Mission of the Department

To establish learning ambience for ushering in computer engineering professionals in core and multidisciplinary area by developing Problem-solving skills through emerging technologies.

Session	2025-26(ODD)	Course Name	Operating System
Semester	5	Course Code	23IOT1504
Roll No	52	Name of Student	Parth Bhedurkar

Practical Number	<b>5</b>
Course Outcome	<ol style="list-style-type: none"> <li>Understand Computer System Configuration and Simulate system resources efficiently using Linux Commands (CO1)</li> <li>Analyse operating system functionalities utilizing system calls, thread programming and process scheduling algorithms (CO2)</li> <li>Apply Synchronization primitives to implement a Deadlock-free solution(CO3)</li> <li>Simulate Disk scheduling, Memory allocation, File allocation, page replacement algorithms (CO4)</li> </ol>
Aim	Stimulate banks deadlock algorithm
Problem Definition	Design and implement a simulation of the <b>Banker's Algorithm</b> for deadlock avoidance. The system should safely allocate resources to multiple processes without entering a deadlock state. The program must check whether a requested resource allocation can be granted without compromising system safety and display the safe sequence if it exists.
Theory (100 words)	<ul style="list-style-type: none"> <li><input type="checkbox"/> Introduction: <ul style="list-style-type: none"> <li>• Page replacement algorithms are used in operating systems to manage memory when a page fault occurs.</li> <li>• When all memory frames are full and a new page needs to be loaded, the system decides which existing page to replace.</li> </ul> </li> <li><input type="checkbox"/> Objective: <ul style="list-style-type: none"> <li>• To reduce the number of page faults and improve overall CPU performance.</li> <li>Efficient page replacement ensures better utilization of physical memory.</li> </ul> </li> <li><input type="checkbox"/> First In First Out (FIFO): <ul style="list-style-type: none"> <li>• This is the simplest page replacement technique.</li> <li>• Pages are replaced in the order they were loaded into memory — the oldest page is replaced first.</li> <li>• It uses a queue to track the order of pages.</li> <li>• Although easy to implement, it may lead to <i>Belady's anomaly</i></li> </ul> </li> </ul>



**Department of Computer Technology B.Tech in Computer Science and Engineering (IOT)**

**Vision of the Department**

To be a well-known centre for pursuing computer education through innovative pedagogy, value-based education and industry collaboration.

**Mission of the Department**

To establish learning ambience for ushering in computer engineering professionals in core and multidisciplinary area by developing Problem-solving skills through emerging technologies.

	<p>(more frames causing more faults).</p> <ul style="list-style-type: none"><li><input type="checkbox"/> Least Recently Used (LRU):<ul style="list-style-type: none"><li>• Replaces the page that has not been used for the longest time. It assumes pages used recently are likely to be used again soon.</li><li>• Implemented using counters or stacks to track recent usage. It gives better performance than FIFO but is more complex.</li></ul></li><li><input type="checkbox"/> Optimal Page Replacement:<ul style="list-style-type: none"><li>• Replaces the page that will not be used for the longest period in the future.</li></ul>It gives the minimum possible page faults but is theoretical as future references are unknown.</li></ul>
Procedure and Execution (100 Words)	<p>Step for Implementation:</p> <p><b>Input the number of processes and resource types.</b></p> <p><b>Input the Allocation matrix</b> (resources currently allocated to each process).</p> <p><b>Input the Maximum matrix</b> (maximum resources each process may need).</p> <p><b>Calculate the Need matrix = Maximum - Allocation.</b></p> <p><b>Input the Available resources.</b></p> <p><b>Run the Banker's safety algorithm:</b></p> <ol style="list-style-type: none"><li>1. Find a process whose needs can be met with available resources.<ul style="list-style-type: none"><li>o Pretend to allocate those resources and mark the process as finished.</li><li>o Release its resources back to available.</li><li>o Repeat until all processes are finished or no further progress is possible.</li></ul></li></ol> <p><b>If all processes finish</b>, the system is in a safe state and the safe sequence is printed.</p> <p><b>If not</b>, the system is in an unsafe state (potential deadlock).</p>

**Vision of the Department**

To be a well-known centre for pursuing computer education through innovative pedagogy, value-based education and industry collaboration.

**Mission of the Department**

To establish learning ambience for ushering in computer engineering professionals in core and multidisciplinary area by developing Problem-solving skills through emerging technologies.

**Code:**

```
#include <stdio.h> #include
<stdbool.h>

int main() { int n,
m;
printf("Enter number of processes: "); scanf("%d",
&n);
printf("Enter number of resource types: ");
scanf("%d", &m);
int alloc[n][m], max[n][m], need[n][m], avail[m];

// Input Allocation Matrix printf("\nEnter
Allocation Matrix:\n"); for (int i = 0; i < n;
i++)
for (int j = 0; j < m; j++)
scanf("%d", &alloc[i][j]);

// Input Maximum Matrix printf("\nEnter
Maximum Matrix:\n"); for (int i = 0; i < n;
i++)
for (int j = 0; j < m; j++)
scanf("%d", &max[i][j]);

// Input Available Resources printf("\nEnter
Available Resources:\n"); for (int j = 0; j < m;
j++)
scanf("%d", &avail[j]);

// Calculate Need Matrix = Max - Allocation for
(int i = 0; i < n; i++)
for (int j = 0; j < m; j++)
need[i][j] = max[i][j] - alloc[i][j];

bool finish[n];
for (int i = 0; i < n; i++) finish[i] =
false;

int safeSequence[n]; int
work[m];
```

**Vision of the Department**

To be a well-known centre for pursuing computer education through innovative pedagogy, value-based education and industry collaboration.

**Mission of the Department**

To establish learning ambience for ushering in computer engineering professionals in core and multidisciplinary area by developing Problem-solving skills through emerging technologies.

```

for (int i = 0; i < m; i++) work[i] = avail[i];
int count=0;

while (count < n) { bool
    found = false;
    for (int i = 0; i < n; i++) { if
        (!finish[i]) {
            bool canRun = true;
            for (int j = 0; j < m; j++) { if
                (need[i][j] > work[j]) {
                    canRun = false; break;
                }
            }
            if (canRun) {
                for (int k = 0; k < m; k++) work[k]
                    += alloc[i][k];

                safeSequence[count++] = i; finish[i] =
                    true;
                found = true;
            }
        }
    }
    if (!found) {
        printf("\nSystem is in UNSAFE state (deadlock may occur).\n");
        return 1;
    }
}

// If system is in a safe state
printf("\nSystem is in SAFE state.\nSafe sequence is: "); for (int
i = 0; i < n; i++)
    printf("P%d ", safeSequence[i]); printf("\n");

return 0;
}

```



**Department of Computer Technology B.Tech in Computer Science and Engineering (IOT)**

**Vision of the Department**

To be a well-known centre for pursuing computer education through innovative pedagogy, value-based education and industry collaboration.

**Mission of the Department**

To establish learning ambience for ushering in computer engineering professionals in core and multidisciplinary area by developing Problem-solving skills through emerging technologies.

	<p>Output:</p> <pre>Enter number of processes: 5 Enter number of resource types: 3  Enter Allocation Matrix: 2 3 5 1 4 3 2 4 6 1 4 6 1 3 2  Enter Maximum Matrix: 7 5 2 4 3 2 4 6 6 8 9 2 5 2 3 6 3  Enter Available Resources: 3 3 2  System is in SAFE state. Safe sequence is: P0 P1 P2 P3 P4  ...Program finished with exit code 0 Press ENTER to exit console.</pre>
Output Analysis	<ul style="list-style-type: none"><li><input type="checkbox"/> The system receives input for <b>5 processes (P0 to P4)</b> and <b>3 resource types</b>.</li><li><input type="checkbox"/> Each process provides its <b>Allocation</b> (resources currently held) and <b>Maximum</b> (maximum resources it may need).</li><li><input type="checkbox"/> The program calculates the <b>Need matrix</b> by subtracting Allocation from Maximum for each resource per process.</li><li><input type="checkbox"/> The system starts with the given <b>Available resources</b>: A = 3, B = 3, C</li></ul>



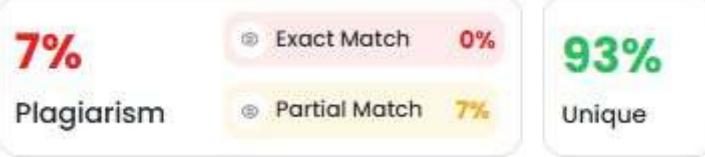
**Department of Computer Technology B.Tech in Computer Science and Engineering (IOT)**

**Vision of the Department**

To be a well-known centre for pursuing computer education through innovative pedagogy, value-based education and industry collaboration.

**Mission of the Department**

To establish learning ambience for ushering in computer engineering professionals in core and multidisciplinary area by developing Problem-solving skills through emerging technologies.

	<p>= 2.</p> <ul style="list-style-type: none"><li><input type="checkbox"/> Using the Banker's Algorithm, it checks if any process's <b>Need</b> can be satisfied with the current <b>Available</b> resources.</li><li><input type="checkbox"/> It successfully finds a <b>safe sequence</b> in which all processes can execute without causing a deadlock.</li><li><input type="checkbox"/> The safe sequence found is: <b>P1 → P3 → P4 → P0 → P2</b>.</li><li><input type="checkbox"/> This means the system is in a <b>SAFE state</b>, and all processes can complete by sequentially allocating and releasing resources.</li><li><input type="checkbox"/> No deadlock will occur if processes request resources as declared in their <b>Maximum matrix</b>.</li></ul>
Link of student Github profile where lab assignment has been uploaded	<a href="https://github.com/parthbhedurkar">https://github.com/parthbhedurkar</a>
Conclusion	The implementation and simulation of the <b>Banker's Algorithm</b> successfully demonstrate how a system can avoid deadlocks by carefully analyzing resource allocation and ensuring it remains in a <b>safe state</b> . By calculating the <b>Need</b> , <b>Available</b> , and using a safety check, the algorithm ensures that all processes can complete without waiting indefinitely for resources. The safe sequence output confirms that resource allocation can proceed without causing a deadlock. This algorithm is crucial in multi-processing environments where resource management and system stability are priorities. Therefore, <b>Banker's Algorithm serves as an effective strategy for deadlock avoidance</b> in operating systems
Plag Report (Similarity index < 12%)	
Date	29/10/2025