# DATA STRUCTURES

**CSE228**

# INITIAL PROJECT REPORT



**Topic:** A program to simulate banker’s algorithm using multithreading.

# Submitted By: Submitted To:

Aryan Pandey Cherry Khosla

K22UP UID: 13436

R.No: 23

**DECLARATION**

I hereby declare that the project work entitled **Banker’s algorithm** is an authentic record of our own work carried out as requirements of Operating system’s Project for the award of B.Tech degree in Computer Science and Engineering (AI & ML) from Lovely Professional University, Phagwara, under the guidance of Waseem Ud Din Wani, during January to May 2023. All the information furnished in this capstone project report is based on our own intensive work and is genuine.

Project Number: 3

Name of Student: ARYAN PANDEY

Registration Number: 12203693

Signature of Student:

ARYAN

Date: 10.23.2023

**CERTIFICATE**

This is to certify that the declaration statement made by this student is correct to the best of my knowledge and belief. He has completed this Capstone Project under my guidance and supervision. The present work is the result of their original investigation, effort and study. No part of the work has ever been submitted for any other degree at any University. The Capstone Project is fit for the submission and partial fulfillment of the conditions for the award of B.Tech degree in Computer Science and Engineering (AI & ML) from Lovely Professional University, Phagwara.

**School of Computer Science and Engineering,**

Lovely Professional University,

Phagwara, Punjab.

**Signature and Name of the Mentor:**

**Designation:**

**ACKNOWLEDGEMENT**

I would like to extend my sincere thanks to the Lovely School of Computer Science and Engineering for providing me with the opportunity to fulfill my wish and achieve my goal. I am grateful to Cherry khosla sir for providing me with the opportunity to undertake this project and for providing me with all the necessary facilities. I am highly thankful to sir for his active support, valuable time and advice, whole-hearted guidance, sincere cooperation, and pain-taking involvement during the study and in completing the assignment of preparing the said project within the time stipulated. Lastly, I am thankful to all those, particularly the various friends, who have been instrumental in creating a proper, healthy, and conducive environment and including new and fresh innovative ideas for me during the project. Without their help, it would have been extremely difficult for me to prepare the project in a timebound framework.

**Introduction**

This comprehensive report outlines the development and implementation of the Banker's Algorithm within a multithreaded environment. The Banker's Algorithm is a pivotal resource allocation and deadlock avoidance technique commonly utilized in operating systems. The primary aim of this project is to construct a dynamic, multithreaded program that simulates the allocation of resources among multiple processes while assuring that resources are distributed in a manner that precludes deadlock occurrences.

**Problem Statement**

The core issue this project addresses is the simulation of resource allocation in a multi-process context. This simulation involves several customers requesting and releasing resources, necessitating that the system remain in a safe state to avoid potential deadlocks.

**Requirement of the Solution**

The logic behind this solution is rooted in the Banker's Algorithm, a well-established method for resource allocation and deadlock avoidance. It operates on the principle of checking if a resource request can be granted without jeopardizing the system's safety. Key steps include:

* Dynamic Thread Creation: To address the dynamic thread creation requirement, we utilize the C++ Standard Library's thread component. Threads are created based on user input, allowing for the simulation of multiple customers dynamically.
* Resource Allocation Simulation: The heart of the project is the simulation of resource allocation. We simulate customers' requests and releases of resources, adhering to the principles of the Banker's Algorithm.
* Safe State Maintenance: The Banker's Algorithm is used to determine if a resource request can be safely granted. It checks if allocating the requested resources will leave the system in a safe state. If not, the request is denied.
* Shared Data Protection: To prevent race conditions and ensure data consistency, mutex locks and condition variables are employed. These synchronization mechanisms ensure that shared data is accessed safely by multiple threads.

**Objective of the project**

This project has several key objectives:

* Dynamic Thread Creation: Threads are created dynamically at runtime, allowing flexibility in handling varying numbers of customers.
* Resource Allocation Simulation: The program simulates resource requests and releases by customers, replicating a real-world scenario.
* Safe State Maintenance: Ensuring that the system remains in a safe state after each allocation, preventing any possibility of deadlock.
* Shared Data Protection: Implementing mutual exclusion using mutex locks to safeguard shared data from concurrent access.

**IMPLEMENTATION**

The implementation is carried out through a series of steps:

* Thread Creation: Customer threads are created dynamically based on user input. This feature provides adaptability to accommodate varying numbers of customers.
* Resource Management: The program simulates customer resource requests and releases, aligning with the principles of the Banker's Algorithm.
* Safety Verification: The Banker's Algorithm is executed to verify that resource requests can be safely granted, ensuring the system remains in a safe state.
* Concurrency Control: Mutex locks and condition variables are employed to control concurrent access to shared data, preventing conflicts and ensuring data consistency.
* User Interaction: The program interacts with the user to input available resources and maximum needs for each customer.

**BENEFITS**

The benefits of this project are multifaceted:

* Dynamic Resource Management: The program dynamically manages resources and customer requests, mirroring real-world resource allocation scenarios where demands may vary over time.
* Deadlock Avoidance: By implementing the Banker's Algorithm, the program ensures that resource allocation remains deadlock-free. This is a critical feature, as deadlocks can lead to system instability and failures.
* Educational Tool: This project serves as a valuable educational tool for individuals interested in understanding resource allocation, multithreading, and synchronization in the context of operating systems.

**PSUEDOCODE**

****

****

****

****

**Source code:** [**banker's algorithm repository Github**](https://github.com/aryanpnd/bankers-algorithm-simulation)

**CONCLUSION**

In conclusion, this project effectively demonstrates the implementation of the Banker's Algorithm within a multithreaded environment. It addresses the critical issue of resource allocation and deadlock avoidance, fulfilling the stated objectives. The robust use of mutex locks and condition variables guarantees that shared data is accessed safely, promoting data integrity. This project provides an insightful and practical example of how operating system principles can be applied to manage resources effectively within a concurrent environment, ensuring system stability and reliability.