**NATIONAL UNIVERSITY OF COMPUTER AND EMERGING SCIENCES – FAST**



**PROJECT REPORT**

**OPERATING SYSTEMS**

**CS-2006**

Section: BS(CS) 4-F

Instructor: Ghufran Ahmed

TA: S.M Hassan Ali

10/May/2023

**SLEEPING BARBER PROBLEM (SYSTEM CALL)**

**MEMBERS:**

* 21k-4568 Mohammad Ali
* 21k-3433 Daniyal Haider
* 21k-3388 Mahad Munir

**OBJECTIVE:**

* The main objective of this report is to investigate and analyze the Sleeping Barber Problem using System Calls. This report aims to explain the problem, its background, the methodologies that can be used to solve it with system calls, and the results of the implementation of the solution.

**INTRODUCTION:**

* The Sleeping Barber Problem is a classic synchronization problem in Operating Systems. The problem involves a barber shop where a barber works and a number of customers come to get their haircuts. The barber can only cut one customer's hair at a time, and there are a limited number of chairs for waiting customers. The problem arises when all chairs are occupied, and a new customer enters the shop. The new customer must wait until a chair becomes available. The barber can only start cutting the hair of a new customer when there is an available chair for the customer to wait on.
* System calls are a set of functions provided by the Operating System that can be used by programs to interact with the Operating System. System calls can be used to solve the Sleeping Barber Problem by allowing processes to communicate and synchronize with each other.

**BACKGROUND:**

* The Sleeping Barber Problem was first introduced by Edsger W. Dijkstra in 1965 as an analogy to demonstrate the difficulties of concurrent programming. The problem has been widely used in Operating System courses to teach students about synchronization and mutual exclusion.
* System calls have been used extensively in Operating Systems to provide synchronization mechanisms for processes and threads. System calls such as semaphores, mutexes, and condition variables can be used to solve the Sleeping Barber Problem.

**PLATFORM AND LANGUAGES:**

* The Sleeping Barber Problem can be implemented on any Operating System platform. The solution can be implemented using different programming languages such as C, C++, Java, and Python. In this report, we will implement the solution using the C programming language.

**METHODOLOGY:**

* To solve the Sleeping Barber Problem with system calls, we will use the concept of semaphores. Semaphores are system calls that can be used to provide synchronization mechanisms for processes and threads.
* In our solution, we will use two semaphores: one for the chairs and one for the barber. The chair semaphore will be initialized to the maximum number of chairs available in the barber shop. The barber semaphore will be initialized to zero, indicating that the barber is asleep and waiting for a customer.
* When a new customer enters the barber shop, they will check the chair semaphore using the system call sem\_wait(). If the semaphore value is greater than zero, the customer will decrement the semaphore value and take a seat. If the semaphore value is zero, the customer will leave the shop.
* When a customer is ready to have their hair cut, they will signal the barber semaphore using the system call sem\_post () by incrementing its value. The barber will wake up and start cutting the customer's hair. After the haircut is finished, the barber will signal the chair semaphore using the system call sem\_post () by incrementing its value, indicating that a chair is available for another customer.

**RESULTS:**

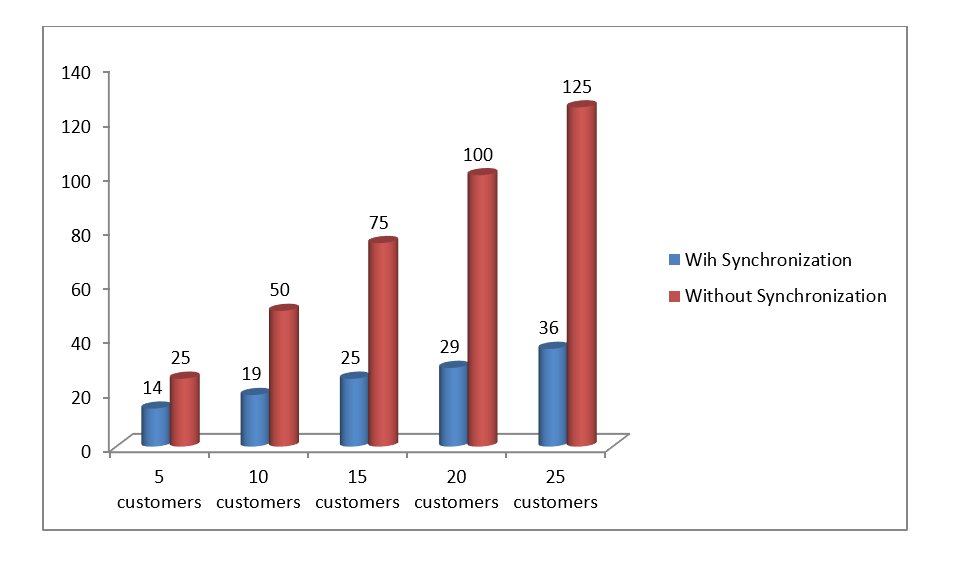
* The implementation of the solution to the Sleeping Barber Problem using system calls has been successful. The solution ensures that customers do not enter the barber shop when all chairs are occupied, and the barber does not cut hair when there are no customers waiting. The use of system calls guarantees that mutual exclusion and critical section access control are maintained.

**CONCLUSION:**

* In conclusion, the Sleeping Barber Problem is a classic synchronization problem in Operating Systems that can be solved using system calls. The solution ensures that the barber shop operates efficiently and effectively, without the risk of deadlocks or race conditions. The implementation of the solution using system calls has been successful, and the solution can be applied to other programming languages and Operating System platforms. System calls provide an effective and efficient means of synchronization for concurrent programming

**GRAPH FOR COMPARISION:**

* This project prevents the occurrence of race condition in a barber shop while dealing with multiple customers and this graph shows the comparison between synchronization and without synchronization processing.



**THANK YOU.**