# Operating Systems

## Assignment 4 Report

**Name: Hanish Goyal (2020MT10805), Harshvardhan Patel (2020MT10808)**

---

- In a concurrent environment, where multiple threads are responsible for operations like insertion, deletion, search, and in-order traversal on the AVL tree, the behavior of the data structure differs from that of a traditional AVL tree, where operations are single-threaded.
The following behavioral differences are observed:

**Write Operations:**

1. **Insertion:** Concurrent insertions lead to race conditions, where multiple threads attempt to modify tree simultaneously. To ensure data consistency and AVL balance, locking mechanisms like mutexes are employed to allow only one thread to modify the tree at a time.
2. **Deletion:** Similar to insertions, concurrent deletions require locking to prevent multiple threads from interfering with each other.

**Read Operations:**

1. **Look Up:** Searches can be safely executed concurrently without major issues. However, to avoid conflicts with concurrent insertions and deletions, any read instruction first waits for all the running write operations. We need to be careful while printing the outputs on the console, as concurrent searches can lead to non determinism in the output as to which search call outputs first. To handle this non determinism, we store the outputs of the search calls and finally print them in a sequential manner.
2. **In-order Traversal:** Similar to search, in order operations can also be performed concurrently with other search operations but not with any in order traversal call.

- **Non Determinism**: Concurrency leads to a non determinism in the final tree compared to a single threaded environment.
- **Performance**: While concurrency can improve the throughput of the AVL tree, it may not always lead to linear speedup due to locking overheads and contention.