

# BIM306 Operating Systems

## Project-2

### Singleton Pattern

Due: June 6, 2021

#### Background

The Singleton pattern ensures a class has only one instance and provides a global point of access to it. One of the pattern implementations is double-checked locking. Implementation of the pattern in object-oriented programming is given:

```
public class Singleton {  
    private volatile static Singleton uniqueInstance;  
  
    private Singleton() {}  
  
    public static Singleton getInstance() {  
        if (uniqueInstance == null) {  
            synchronized (Singleton.class) {  
                if (uniqueInstance == null) {  
                    uniqueInstance = new Singleton();  
                }  
            }  
        }  
        return uniqueInstance;  
    }  
}
```

Check for an instance and if there isn't one, enter a synchronized block.

Note we only synchronize the first time through!

Once in the block, check again and if still null, create an instance.

\* The volatile keyword ensures that multiple threads handle the uniqueInstance variable correctly when it is being initialized to the Singleton instance.

! If performance is an issue in use of getInstance() method then this method can drastically reduce the overhead.

#### Project Definition

In the project, first, you are asked to implement the getInstance method according to the double-checked locking using C programming language (POSIX library) to the given template code. Also, you should use semaphore. This will not be the same as the object-oriented version naturally, but the idea is the same.

Second, you should explain your code how to implement it step by step.