

- Exercise

In this lesson, there is some fun 'try it outs' for you to develop a better understanding of mutexes.

WE'LL COVER THE FOLLOWING



- Try It Out! - No Synchronization
- Try it Out! - How Smart is Your C++ Runtime?

Try It Out! - No Synchronization

Let the program below *write* to `std::cout` without synchronization and observe its output.

```
//mutex.cpp
#include <chrono>
#include <iostream>
#include <mutex>
#include <string>
#include <thread>

std::mutex coutMutex;

class Worker{
public:
    explicit Worker(const std::string& n):name(n){};

    void operator() (){
        for (int i= 1; i <= 3; ++i){
            // begin work
            std::this_thread::sleep_for(std::chrono::milliseconds(200));
            // end work
            //coutMutex.lock();
            std::cout << name << " : " << "Work " << i << " done !!!" << std::endl;
            //coutMutex.unlock();
        }
    }
private:
    std::string name;
};

int main(){
```

```

std::cout << std::endl;

std::cout << "Boss: Let's start working." << "\n\n";

std::thread herb= std::thread(Worker("Herb"));
std::thread andrei= std::thread(Worker("  Andrei"));
std::thread scott= std::thread(Worker("    Scott"));
std::thread bjarne= std::thread(Worker("      Bjarne"));
std::thread andrew= std::thread(Worker("        Andrew"));
std::thread david= std::thread(Worker("          David"));

herb.join();
andrei.join();
scott.join();
bjarne.join();
andrew.join();
david.join();

std::cout << "\n" << "Boss: Let's go home." << std::endl;

std::cout << std::endl;
}

```



Try it Out! - How Smart is Your C++ Runtime?

Locking a non-recursive mutex more than once is undefined behavior. Run the code in the example below to see how this works.

```

#include <mutex>
#include <thread>

```



```

int main(){
    std::mutex m;

    m.lock();
    m.lock();

}

```



For further information, see the following:

- [std::mutex](#)

- `std::timed_mutex`
 - `std::recursive_mutex`
 - `std::recursive_timed_mutex`
 - `std::shared_timed_mutex`
-

In the next lesson, we will learn about locks in modern C++.