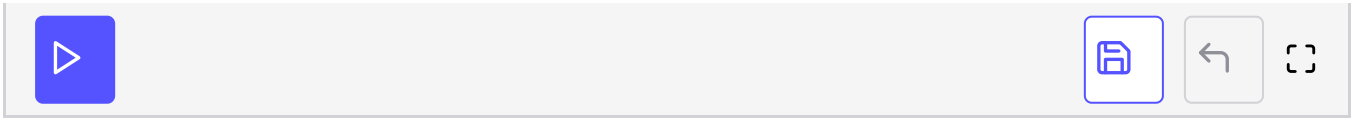
Thread Local Summation: Using Thread Local Data

This lesson explains the solution for calculating the sum of a vector problem using thread local data in C++.

Thread-local data belongs to the thread in which it was created; it will only be created when needed. Thread-local data is an ideal fit for the local summation variable `tmpSum`.

```
// threadLocalSummation.cpp

#include <atomic>
#include <chrono>
#include <iostream>
#include <random>
#include <thread>
#include <utility>
#include <vector>

constexpr long long size = 100000000;

constexpr long long fir =  25000000;
constexpr long long sec =  50000000;
constexpr long long thi =  75000000;
constexpr long long fou = 100000000;

thread_local unsigned long long tmpSum = 0;

void sumUp(std::atomic<unsigned long long>& sum, const std::vector<int>& val,
           unsigned long long beg, unsigned long long end){
    for (auto i = beg; i < end; ++i){
        tmpSum += val[i];
    }
    sum.fetch_add(tmpSum, std::memory_order_relaxed);
}

int main(){

  std::cout << std::endl;

  std::vector<int> randValues;
  randValues.reserve(size);

  std::mt19937 engine;
  std::uniform_int_distribution<> uniformDist(1, 10);
  for (long long i = 0; i < size; ++i)
      randValues.push_back(uniformDist(engine));

  std::atomic<unsigned long long> sum{};
  const auto sta = std::chrono::system_clock::now();
```

```
  std::thread t1(sumUp, std::ref(sum), std::ref(randValues), 0, fir);
  std::thread t2(sumUp, std::ref(sum), std::ref(randValues), fir, sec);

  std::thread t3(sumUp, std::ref(sum), std::ref(randValues), sec, thi);
  std::thread t4(sumUp, std::ref(sum), std::ref(randValues), thi, fou);

  t1.join();
  t2.join();
  t3.join();
  t4.join();

  const std::chrono::duration<double> dur=
        std::chrono::system_clock::now() - sta;

  std::cout << "Time for addition " << dur.count()
            << " seconds" << std::endl;
  std::cout << "Result: " << sum << std::endl;

  std::cout << std::endl;

}
```



I declare the thread local variable `tmpSum` in line 18 and use it for the addition in lines 23 and 25.

In the next lesson and the last scenario, I will use tasks.