

МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ
УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ
“БРЕСТСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ”
ИНТЕЛЛЕКТУАЛЬНЫЕ ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ

ОТЧЁТ
по лабораторной работе №5
“ Многопоточность.”

Выполнил:
студент группы ПО-9
Сольшко Д.А.

Проверил:
Козик И.Д.

Брест 2023

Лабораторная работа №5

Вариант №19

Цель работы: научиться работать с многопоточностью в приложениях.

Основное содержание работы.

Необходимо написать 2 программы, используя в них несколько потоков. Одну из программ реализовать через атомные переменные, вторую – через mutex. Отчёт должен содержать цель работы, код программы, скриншот работы программы и вывод по лабораторной работе.

Задание: 3. Создать три потока, выполняющих различные арифметические операции над переменной типа float.

Текст программы:

Программ с использованием атомной переменной:

```
#include <iostream>
#include <thread>
#include <atomic>
#include <mutex>

using namespace std;

int main()
{
    setlocale(LC_ALL, "ru");

    atomic<float> counter(0);
    thread thread1([&](){
        for (int i = 0; i < 100; ++i) {
            counter.store(counter.load() + 2);
        }
    });

    thread thread2([&](){
        for (int i = 0; i < 100; ++i) {
            counter.store(counter.load() - 1.0);
        }
    });

    thread thread3([&](){
        for (int i = 0; i < 10; ++i) {
            counter.store(counter.load() * 1.3);
        }
    });

    thread1.join();
    thread2.join();
    thread3.join();

    cout << "Значение переменной после операций " << counter.load() << endl;

    return 0;
}
```

Программа с использованием мьютекса:

```
#include <iostream>
#include <mutex>
#include <thread>

using namespace std;

int main() {
```

```

setlocale(LC_ALL, "ru");
float counter = 0;
mutex counterMutex;
thread thread1([&](){
    for (int i = 0; i < 100; ++i) {
        counterMutex.lock();
        counter += 2;
        counterMutex.unlock();
    }
});

thread thread2([&](){
    for (int i = 0; i < 100; ++i) {
        counterMutex.lock();
        counter -= 1;
        counterMutex.unlock();
    }
});

thread thread3([&](){
    for (int i = 0; i < 10; ++i) {
        counterMutex.lock();
        counter *= 1.3;
        counterMutex.unlock();
    }
});

thread1.join();
thread2.join();
thread3.join();


cout << "Значение переменной после операций " << counter << endl;

return 0;
}

```

Результаты программы:

Программа с использованием атомной переменной:


 Консоль отладки Microsoft Visual Studio

Значение переменной после операций 1378.58

Программа с использованием мьютексов:

$1.3^{10} \times 100 =$

1378.58491849

 Консоль отладки Microsoft Visual Studio

Значение переменной после операций 1378.58

Вывод: как итог полученное значение от потоков соответствует значению, полученному самостоятельно с использованием калькулятора. Я научился работать с многопоточностью в приложениях.