Семинар #3: Синхронизация. Домашнее задание.

Задача 1. Два потока с `std::condition_variable`

Напишите программу, в которой вам нужно будет создать два потока:

- Первый поток будет считывать текстовый файл построчно и записывать каждую строку из файла поочерёдно в глобальную переменную типа `std::string`. После того как поток записал строку в глобальную переменную, он должен сообщить об этом другому потоку с помощью условной переменной и ждать сообщение от второго потока.
- Второй поток должен ждать пока в глобальную строку что-то запишется (строка перестанет быть пустой), используя условную переменную. После этого он должен брать эту строку, превращать все строчные буквы в этой строке в заглавные и печатать строку на экран. После этого он должен делать строку пустой, сообщать об этом первому потоку и ждать сообщение от первого потока.

Протестируйте вашу программу на файле `invisible_man.txt`.

Задача 2. Поиск максимума, используя `std::async`

В файле `code/00problem_paralel_max.cpp` написана функция:

```
uint64_t getMax(const std::vector<uint64_t>& v)
```

которая принимает на вход вектор чисел и возвращает максимальный элемент в этом векторе.
Напишите функцию:

```
uint64_t getMax(int n, const std::vector<uint64_t>& v)
```

которая будет делать то же самое, но параллельно, используя `n` потоков.
Для распараллеливания используйте `std::async`.
Протестируйте функцию, замерив скорость работы однопоточной и многопоточной версии.

Задача 3. Поиск максимума, используя `std::packaged_task`

Напишите функцию:

```
uint64_t getMax(int n, const std::vector<uint64_t>& v)
```

которая будет делать то же самое, но для распараллеливания используйте `std::packaged_task`.
Протестируйте функцию, замерив скорость работы однопоточной и многопоточной версии.

Задача 4. Средний цвет изображений

Напишите программу, которая будет в цикле считывать строки. Каждая строка является именем изображения в формате `.ppm`. Программа должна вычислять "средний цвет" изображения, то есть среднее арифметическое каждой компоненты цвета по всем пикселям. После вычисления программа должна печатать этот средний цвет. Программа должна заканчиваться когда считывается строка "quit". Примерный вывод программы должен выглядеть следующим образом:

```
Enter image name: house.ppm
Enter image name: big.ppm
Average color of house.ppm is: (50, 100, 200)
Enter image name: small.ppm
Enter image name: no
File no do not exist
Average color of small.ppm is: (200, 100, 200)
Average color of big.ppm is: (150, 200, 100)
Enter image name: quit
```

Программа должна работать асинхронно, то есть во время когда вы вводите название второго файла, программа должна обрабатывать первый файл в другом потоке. Для считывания файла в формате `.ppm` используйте класс изображения из файла `code/image.hpp`. Изображения можно найти тут: `cs_mipt_faki/data/ppm_images`.