МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ «ЛЭТИ»
ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МО ЭВМ

ОТЧЁТ

по лабораторной работе №2
по дисциплине «Параллельные алгоритмы»
Тема: Реализация взаимодействия потоков по шаблону
“производитель-потребитель”

Студент гр. 9303 _____ Эйсвальд М.И.

Преподаватель _____ Сергеева Е.И.

Санкт-Петербург
2022

Цель работы.

Научиться синхронизировать потоки, изучить шаблон взаимодействия потоков «производитель-потребитель».

Задача.

На базе лаб. 1 (части 1.2.1 и 1.2.2) реализовать итерационное (потенциально бесконечное) выполнение подготовки, обработки и вывода данных. Обеспечить параллельное выполнение потоков обработки готовой порции данных, подготовки следующей порции данных и вывода предыдущих полученных результатов.

Выполнение работы.

В лабораторной работе использован код предыдущей лабораторной работы: класс `Matrix` и функция предварительной настройки по аргументам командной строки.

Для синхронизации процессов был создан класс `SyncBuffer`, который содержит код, отвечающий за синхронизированное добавление элемента в буфер и синхронизированное удаление элемента из буфера. Для синхронизации используются `std::conditon_variable`. Для хранения данных `SyncBuffer` использует массив и два указателя: на следующую позицию, с которой можно читать, и на следующую позицию в массиве, с которой можно писать. При такой структуре полный и пустой буфер выглядят одинаково: указатели находятся на одной позиции. Поэтому дополнительно введена переменная `empty`, которая указывает, пустой ли сейчас буфер или нет. Количество заполненных позиций в буфере явно не хранится.

В программе одновременно существуют три буфера: два буфера с аргументами для сложения (первый и второй аргумент для каждой итерации сложения находятся в разных буферах) и буфер, куда заносятся результаты,

ожидающие печати в файл. С буферами работают три потока: `generator`, `summator` и `printer`. Первый генерирует матрицы и заносит их в буферы для аргументов; второй достаёт сгенерированные матрицы из буферов, создаёт n дочерних потоков (n указывается при запуске программы), которые складывают различные элементы матриц таким же образом, как и в лабораторной работе №1, ждёт их завершения, потом добавляет полученную матрицу в буфер ожидающих печати матриц; третий выводит результаты сложения из буфера в файл.

Демонстрация работы программы представлена на рисунке ниже.

```
michael@michael-TUF-Gaming-FX505DT-FX505DT:~/Desktop/work/PA/labs_PA_22/9303_Eiswald_Mikhail$ ./build-lb2-Desktop_Qt_5_15_2_GCC_64bit-Debug/lb2 11 2 2 10 10
Generating and adding 2x2 matrices with elements up to 10 using 10 threads 11 times
Done. Check output.txt for results.
michael@michael-TUF-Gaming-FX505DT-FX505DT:~/Desktop/work/PA/labs_PA_22/9303_Eiswald_Mikhail$ cat output.txt
/           1          -6 \
\           0           0 /
/          -2         -11 \
\          13         -17 /
/          11          -9 \
\           4          -6 /
/          -7           9 \
\          10          19 /
/           1           2 \
\           7          12 /
/           5         -14 \
\          -2           4 /
/           5          12 \
\         -17          -1 /
/         -15          -7 \
\         -14          12 /
/          13          -8 \
\           4           2 /
/           9          -9 \
\          -1          -1 /
/          13           6 \
\           1           8 /
```

Рисунок 1 – Демонстрация работы программы

Вывод.

В ходе выполнения работы были изучены принципы синхронизации потоков, был изучен шаблон взаимодействия потоков «производитель-потребитель». Результатом работы стала программа, обрабатывающая данные с использованием шаблона «производитель-потребитель», использующая `std::conditon_variable` для синхронизации.