**Домашнее задние №5**

**(Архитектура вычислительных систем)**

**Вариант 25**

Выполнил

Пименов Михаил

БПИ208

**Задача о производстве булавок.**

В цехе по заточке булавок все необходимые операции осуществляются тремя рабочими. Первый из них берет булавку и проверяет ее на предмет кривизны. Если булавка не кривая, то рабочий передает ее своему напарнику. Напарник осуществляет собственно заточку и передает заточенную булавку третьему рабочему, который осуществляет контроль качества операции. Требуется создать многопоточное приложение, моделирующее работу цеха. При решении использовать парадигму «производитель-потребитель»

**Краткое описание парадигмы «производитель-потребитель»**

Производители и потребители – это парадигма взаимодействующих неравноправных потоков. Одни потоки «производят» данные, другие их «потребляют». Часто такие потоки организуются в **конвейер**, через который проходит информация. Каждый поток конвейера потребляет выход своего предшественника и производит входные данные для своего последователя.

Другой распространенный способ организации потоков – древовидная структура или сети слияния, на этом основан, в частности, метод **дихотомии**.

**Реализация**

Для реализации данной задачи буду использовать парадигму «производитель-потребитель», у меня будет один класс для чистого производителя – станка, который производит болванки для булавок (Machine), и 3 класса для потребителей:

* ConsumerStraightChecker – проверяет кривизну
* ConsumerSharpener – производит заточку
* ConsumerAllChecker – осуществляет контроль

Они будут осуществлять проверки. Но они также будут являться и производителями, так как после проверки они будут направлять данные дальше.

Я сделал 4 асинхронных буфера - очереди (написав класс AsyncBuffer c использованием mutex, condition\_variable, lock\_guard, unique\_lock).

1. Machine кладет Pin в buffer\_new\_pins (вероятность получить кривую булавку 30%).
2. ConsumerStraightChecker берет Pin из buffer\_new\_pins, проверяет, и в случае корректности кладет Pin в buffer\_straight\_pins.
3. ConsumerSharpener берет Pin из buffer\_straight\_pins, затачивает её (вероятность успешной заточки 90%), кладет её в buffer\_sharpened\_pins
4. ConsumerAllChecker берет Pin из buffer\_sharpened\_pins и в случае, если всё корректно, кладёт ее в buffer\_checked\_pins.

Базовое понимание парадигмы получил из <https://intuit.ru/studies/courses/641/497/lecture/11282>, дальше додумал сам для этой задачи.

Для получения и запуска бинарного файла нужно выполнить следующие команды в папке task5, предварительно очистив папку build:

cmake -B ./build

cd ./build

make

./task5