Лабораторная работа: RISC-V тулчейн на примере матриц

- Напишите перемножение матриц на языке C. В качестве алгоритма можете выбрать любой, в том числе и тривиальную вещь. Дополнительное задание — реализуйте cache friendly, векторезуемый алгоритм перемножения
- Ваша программа должна быть написана с учетом хороших практик — читабельность, расширяемость, удобство интерфейсов, ООП. Также должны быть написаны тесты с возможностью удобного добавления
- Для анализа вашей программы должны быть использованы **valgrind, gprof, address sanitizer** под x86_64. Дополнительное задание — заставить по возможности работать санитайзеры и valgrind с RISC-V (не известно, насколько они портированы, советую делать это доп задание последним)
- Ваша программа должна поддерживать отладку как хостовым дебаггером, так и дебаггером под RISC-V (отладка в qemu)
- Ваша программа должна содержать базовую систему сборки (make или cmake). Программа должна собираться как под x86_64, так и под RISC-V с учетом особенностей сборки (нужен кросс тулчейн)
- На x86_64 ваша программа тестируется нативно, в то время как на RISC-V тестирование должно быть организовано с помощью **qemu-riscv64**.
- Все инструменты для разработки под RISC-V брать из **Syntacore Devtoolkit**
- Ваша программа должна иметь возможность запускать тесты нативно (если вы находитесь на системе с RISC-V процессором)
- Дополнительное задание — вам необходимо написать несколько тестов — бенчмарков, задача которых - анализировать вашу программу (алгоритм) на производительность. При доступе к плате эти тесты нужно будет прогнать.
- Альтернативно (или как плюс еще одно дополнительное задание) реализовать бенчмаркинг вашей программы на симуляторе **GEM5**
- По каждому пункту можно и нужно задавать вопросы, не стесняемся. Даже если не понятно, как подступить к конкретному пункту. Так же очень советую гуглить и читать туториалы и документации по инструментам.