Архитектура вычислительных систем.

Вариант 20

Работа № 5

Выполнил:  
Лобанович Михаил Михайлович,  
группа БПИ-208

**Условие задачи:**

*Задача о программистах*. В отделе работают три программиста. Каждый программист пишет свою программу и отдает ее на проверку другому программисту. Программист проверяет чужую программу, когда его собственная уже написана. По завершении проверки, программист дает ответ: программа написана правильно или написана неправильно. Программист спит, если не пишет свою программу и не проверяет чужую программу. Программист просыпается, когда получает заключение от другого программиста. Если программа признана правильной, программист пишет другую программу, если программа признана неправильной, программист исправляет ее и отправляет на проверку тому же программисту, который ее проверял. Создать многопоточное приложение, моделирующее работу программистов.

**Дополнительное условие:**

Так как в условии задачи не описана ситуация, в которой приложение завершают работу, мною было введено число задач для каждого программиста, после корректного выполнения которых программа завершает работу. Ограничение на данное число для каждого из программистов – от 0 до 100.

**Особенности реализации:**

Каждый программист представлен отдельным потоком. Каждый из них при наличии возможности пишет задачу, проверяет чужую, если очередь для проверки не пуста, а затем засыпает на 500 миллисекунд. Сон повышает производительность программы, так как программисты, которым нечего писать и проверять в это время не будут работать.

Написание программы представляет из себя генерацию числа от 0 до 9 и последующей отправки другому случайному (вероятности равны) программисту пары чисел – индекса программиста, написавшего задачу и само сгенерированное число. После этого у программиста, написавшего программу нет возможности писать новую, пока старую не проверят, однако он может проверять чужие работы. Без этого функционала программа могла бы уйти в бесконечное ожидание.

Проверка программы представляет из себя определение программиста, приславшего задачу, а затем проверку на четность присланной «программы». Считается, что программа выполнена верна, если она четна. В таком случае общее количество оставшихся задач уменьшает, иначе количество задач остается прежним, так как программу необходимо переделать. После этого у программиста, чью задачу проверили, вновь появляется возможность писать программу.

Перед действием каждого программиста «закрывается» мьютекс, так как у программистов много общих переменных, с которыми они работают. Это необходимо, чтобы исключить доступ к одним и тем же переменным разными потоками и таким образом избежать ошибок, связанных с неопределенным поведением.

Все действия с указанием номера программиста выводятся в консоль.

**Выбор парадигмы параллельного программирования:**

Мной была выбрана парадигма взаимодействующие равные. Описание парадигмы:

Взаимодействующие равные – модель, в которой исключен не занимающийся непосредственными вычислениями управляющий поток.

Распределение работ в таком приложении либо фиксировано заранее, либо динамически определяется во время выполнения. Одним из распространенных способов динамического распределения работ является «портфель задач». Портфель задач, как правило, реализуется с помощью разделяемой переменной, доступ к которой в один момент времени имеет только один процесс.

Вычислительная задача делится на конечное число подзадач. Как правило, каждая подзадача должна выполнить однотипные действия над разными данными. Подзадачи нумеруются, и каждому номеру определяется функция, которая однозначно отражает номер задачи на соответствующий ему набор данных. Создается переменная, которую следует выполнять следующей. Каждый поток сначала обращается к портфелю задач для выяснения текущего номера задачи, после этого увеличивает его, потом берет соответствующие данные и выполняет задачу, затем обращается к портфелю задач для выяснения следующего номера задачи.

Естественно, должен быть предусмотрен механизм остановки процессов при исчерпывании всего множества задач, как в «производителях и потребителях».

То есть поток получает задачу из портфеля и пока задача остается не выполненной, поток ее решает, а затем снова получает задачу из портфеля.

Так как все три программиста равны, постоянно выполняют вычисления (даже если программист не пишет свою программу, он продолжает проверять чужие), при этом работа выполняется динамически при помощи распределяемых переменных.

**Ввод:**

Три целых числа от 0 до 100. Для каждого программиста в консоль выводится соответствующее сообщение с подсказкой.

**Дополнительные источники:**

<https://pro-prof.com/forums/topic/parallel-programming-paradigms> - дополнительное изучение информации о парадигмах параллельного программирования, среди которых выбранная мной