Задание состоит из двух программ, которые необходимо реализовать. Взаимодействие программ должно быть реализовано через использование сокетов.

**Программа №1.**

Должна состоять из двух потоков и одного общего буфера.

Поток 1. Принимает строку, которую введет пользователь. Должна быть проверка, что строка состоит только из цифр и не превышает 64 символа. После проверки строка должна быть отсортирована по убыванию и все элементы, значение которых чётно, заменены на латинские буквы «КВ». После данная строка помещается в общий буфер и поток должен ожидать дальнейшего ввода пользователя.

Поток 2. Должен обрабатывать данные которые помещаются в общий буфер. После получения данных общий буфер затирается.

Поток должен вывести полученные данные на экран, рассчитать общую сумму всех элементов , которые являются численными значениями. Полученную сумму передать в Программу №2. После этого поток ожидает следующие данные.

*Примечание №1 по Программе №1: Взаимодействие потоков должно быть синхронизировано, поток №2 не должен постоянно опрашивать общий буфер. Механизм синхронизации не должен быть глобальной переменной.*

*Примечание №2 по Программе №1: Работа программы должна быть максимально независима от статуса запуска программы №2. Это значит, что внезапный останов программы №2 не должен приводить к немедленным проблемам ввода у пользователя.*

*При перезапуске программы №2 необходимо произвести передподключение.*

**Программа №2.**

Ожидает данные от Программы №1. При получении данных происходит анализ из скольки символов состоит переданное значение. Если оно больше 2-ух символов и если оно кратно 32 выводит сообщение о полученных данных, иначе выводится сообщение об ошибке. Далее программа продолжает ожидать данные.

*Примечание №1 по Программе №2: Работа программы должна быть максимально независима от статуса запуска программы №1.*

*Внезапный останов программы №1 не должен приводить к немедленным проблемам отображения. Необходимо ожидать подключение программы №1 при потере связи между программами.*

*Примечание по заданию: Не обязательно все размещать в одном классе. Может быть разработана иерархия классов. Чем более функционален интерфейс класса, тем лучше.*

**Требования к присылаемым решениям.**

* Готовое задание должно быть передано ответным письмом в zip-архиве. Ссылки на личные хранилища и файлообменники не принимаются.
* Исходные коды каждой программы должны находиться в своей директории.
* Программа должна собираться под ОС Linux и не требовать установки дополнительных пакетов, библиотек, изменений сборочных файлов, нахождения определённых файлов в специфичном месте и т.п. Приветствуются readme-файлы с описанием шагов сборки и работы программы.
* Исходный код должен компилироваться средствами CMake или Make с использованием компилятора GCC.
* В директориях с исходным кодом не должно быть мусора: неиспользуемых файлов исходных кодов или ресурсов, промежуточных файлов сборки (в том числе генерируемых системой сборки), бинарных файлов и т.п.
* Максимальное время на выполнение задания - 1 неделя.