Задача:

Требуется реализовать утилиту, работающую из командной строки и выполняющую сканирование файлов в заданной директории, с целью нахождения в ней “вредоносных” файлов.

Функциональные требования

- Путь к корневой директории для проверки передается на вход утилиты в качестве параметра
- Проверка должна быть рекурсивной, с заходом во вложенные директории
- “Вредоносность” файла определяется путем подсчета MD5-хеша от его содержимого и поиска этого хеша среди заданных
- После завершения выполнения утилиты пользователю в консоль должен быть выведен отчет о сканировании, в котором присутствует следующая информация:
    - Общее количество обработанных файлов
    - Количество обнаруженных “вредоносных” файлов
    - Количество ошибок анализа файлов (например, не хватает прав на чтение файла);
    - Время выполнения утилиты
- В процессе проверки утилита должна логгировать информацию об обнаруженных “вредоносных” файлах
    - Минимально логгируемая информация:
        - Путь к файлу
        - Хеш
        - Вердикт
    - В качестве лога использовать текстовый файл, путь к которому также передается на вход утилите
- Размеры файлов могу превышать объем доступной оперативной памяти

База вредоносных хэшей

Хеши “вредоносных” файлов и соответствующие им вердикты задаются в CSV-файле, путь к которому передается на вход утилите.
CSV-файл – это текстовый файл, состоящий из строк. Каждая строка содержит хеш и имя вердикта, разделенные символом ';'.

Пример содержимого csv-файла (конкретные значения хешей и вердиктов можно вбить самостоятельно):

```
a9963513d093ffb2bc7ceb9807771ad4;Exploit
ac6204ffeb36d2320e52f1d551cfa370;Dropper
8ee70903f43b227eeb971262268af5a8;Downloader
```

Пример запуска

```
scanner.exe --base base.cvs --log report.log --path c:\folder
```

Требования к реализации

- Реализовать утилиту необходимо на языке C++ с использованием ООП, под Windows (возможна кросс-платформенная реализация)
- Использовать CMake в качестве системы сборки
- В процессе работы утилиты задействовать все доступные вычислительные ресурсы системы
- Архитектурно утилита должна состоять из двух компонентов:
    - исполняемого файла, предоставляющего консольный интерфейс для ввода команды на сканирование и вывода финального отчета
    - DLL, в которой расположена основная логика задачи сканирования
- Написать юнит тесты на основной функционал утилиты (в качестве тестового фреймворка использовать gtest).

Загрузка решения

- Все файлы решения (вместе с исходниками) загрузите одним архивом с названием ИМЯ_ФАМИЛИЯ_C++ в наше хранилище: **https://box.kaspersky.com/u/d/a02d3b2ec47742a68495/**
- Когда закончишь выполнение, вернись на страницу откуда ты выгружал это задание, подгрузи файл в поле ответа со словом «Выполнено» внутри.