Пояснительная записка

ФИО, группа

Шерстюгина Анастасия Андреевна, БПИ204

Описание полученного задания (вариант 32):

Пляшущие человечки. На тайном собрании глав преступного мира города Лондона председатель собрания профессор Мориарти постановил: отныне вся переписка между преступниками должна вестись тайнописью. В качестве стандарта были выбраны «пляшущие человечки», шифр, в котором каждой букве латинского алфавита соответствует хитроумный значок. Реализовать многопоточное приложение, шифрующее исходный текст (в качестве ключа используется кодовая таблица, устанавливающая однозначное соответствие между каждой буквой и каким-нибудь числом). Каждый поток шифрует свои кусочки текста. При решении использовать парадигму портфеля задач.

Структура проекта:

avs/
- cmake-build-debug/
- test/
- main.cpp
- dispatcher.h
- dispatcher.cpp
- cryptographer.h
- cryptographer.cpp

Работа программы:

Пользователь может задать входные данные с помощью команды:

count *length* *input path*

count - количество потоков-шифровальщиков

length - длина строки, которую обрабатывает каждый поток-шифровальщик

input path - путь к текстовому файлу, в котором лежит текст для шифрования

Если пользователь желает сгенерировать текст для шифрования, то вместо *input path* ему следует написать GENERATE.

По окончании многопоточной обработки текста программа сохранит результат в файле *input path*_out, а также для проверки будет сохранен результат шифрования без использования многопоточности в файле *input path*_expected.

Для демонстрации работы в папке test лежат примеры, для которых использовались следующие команды:

```
./project 5 1000 ../test/example
./project 20 100 GENERATE
```

Основные характеристики программы:

- Число интерфейсных модулей = 2
- Число модулей реализации = 3
- Общий размер исходных текстов ≈ 12.1 КБ
- Общий размер исполняемых файлов ≈ 701.2 КБ

Инструментальные средства:

- Виртуальная машина Oracle VM VirtualBox
  - Класс ОС: Linux
  - Версия ОС: Ubuntu (64-bit)
  - Кол-во процессоров виртуальной машины: 2
  - Оперативная память: 4096 МБ
- Языки программирования: C++
- Библиотеки: iostream, string, cstring, thread, ctime, vector, fstream, map, mutex
- Интегрированная среда разработки: CLion
- Средства сборки проектов: CMake

Принцип работы:

Главные сущности в программе – это диспетчер (объект класса) и шифровальщики (потоки).

Задачи диспетчера – последовательно раздавать куски текста шифровальщикам и хранить результат шифрования. Может возникнуть сложность, когда несколько шифровальщиков одновременно обращаются к диспетчеру за очередным куском текста, то есть претендуют на один и тот же ресурс, который будем называть разделяемой переменной. Так как каждый шифровальщик должен обрабатывать уникальный кусок текста, диспетчер раздает их строго по очереди поступления запросов. Реализуется это с помощью мьютекста, который блокируется, когда поток обратился за выдачей куска текста, и освобождается, когда поток текст получил. Диспетчер заканчивает свою работу, когда весь текст разобрали (=зашифровали).
Получаем, что диспетчер реализует портфель задач.

Шифровальщик работает следующим образом – обращается к диспетчеру за кусочком текста, если получает его – обрабатывает и возвращает диспетчеру в зашифрованном виде, повторяет этот процесс. Если не получает текст, значит, что текста больше нет, диспетчер закончил свою работу и шифровальщик тоже заканчивает свою работу.

Обратим внимание, что запись зашифрованного результата может осуществляться одновременно, так как каждый поток записывает строго свой кусок, они не пересекаются и не претендуют на одни и те же ресурсы. Реализуется это с помощью хранения индекса начала куска текста (относительно всего текста), по этому же индексу в дальнейшем и записывается зашифрованный результат. А так как куски выдаются строго по очереди (как описывалось выше), то и при записи результатов не возникнет пересечений.

Программа считает время выполнения многопоточного шифрования, а также сравнивает полученный результат с шифрованием без использования потоков.

Ключ для шифрования – пары <символ, код>, где для каждой строчной и заглавной буквы латинского алфавита генерируется уникальное целое число от 0 до 999. Результат шифрования – массив целых чисел.