Отчет по работе кода для шифрования, используя метод SHA-384
Теоретическая часть

SHA-384 — это криптографическая хеш-функция семейства SHA-2. Она используется для проверки целостности данных, хранения паролей в хешированном виде, создания цифровых подписей и других задач, где требуется надёжная необратимая функция.

Хеш-функции широко применяются в информационной безопасности:

1. для проверки целостности данных;
2. в цифровых подписях;
3. для безопасного хранения паролей;
4. при передаче сообщений по незащищенным каналам.

Функция SHA-384 обладает следующими ключевыми свойствами:

1. **Однонаправленность**: невозможно восстановить исходное сообщение по хешу;
2. **Устойчивость к коллизиям**: крайне сложно найти два разных сообщения с одинаковым хешем;
3. **Лавинный эффект**: малейшее изменение входного сообщения приводит к полному изменению хеш-кода.

Описание работы программы

Программа реализует вычисление криптографического хеша SHA-384 для произвольной строки, введенной пользователем с клавиатуры. Она написана на языке C++ и использует библиотеку OpenSSL для выполнения хеширования.

Основные этапы работы:

1. При запуске программа запрашивает у пользователя ввод строки.
2. Введенная строка передается в специальную функцию, которая вычисляет её SHA-384 хеш.
3. Хеш преобразуется в строку шестнадцатеричного формата (читаемый вид), и выводится на экран.

Как происходит вычисление хеша:

1. Используется стандартная функция SHA384 из библиотеки OpenSSL, которая принимает строку (в виде массива байтов) и возвращает массив байтов, содержащий хеш.
2. Далее хеш переводится в строку шестнадцатеричных символов, по два символа на байт, чтобы результат можно было отобразить пользователю.
3. Итоговая строка имеет фиксированную длину — 96 символов (так как SHA-384 даёт 384 бита = 48 байт, по 2 hex-символа на байт).

Пример результата, используя докер для компиляции:

```
shrimp@Alexandras-MacBook-Pro sha384_service % docker build -t sha384_app .
[docker run -it sha384_app
[+] Building 553.6s (10/10) FINISHED                                docker:desktop-linux
 => [internal] load build definition from Dockerfile                                0.0s
 => => transferring dockerfile: 476B                                                0.0s
 => [internal] load metadata for docker.io/library/gcc:13                           2.7s
 => [internal] load .dockerignore                                                   0.0s
 => => transferring context: 2B                                                     0.0s
 => [1/5] FROM docker.io/library/gcc:13@sha256:d9aefdb57822bfa4738c8223c6e43ca  538.9s
 => => resolve docker.io/library/gcc:13@sha256:d9aefdb57822bfa4738c8223c6e43ca2b  0.0s
 => => sha256:04c56742fba5472466f00cc7c39c3228cd6c4717e5aa9837ee 9.60kB / 9.60kB  0.5s
 => => sha256:ff68f4c56b7dc37116652db4942b512862937663c77d6a3ac4 1.80kB / 1.80kB  0.3s
 => => sha256:743f355a2ec271a051da139fb32e50f022c75db753d5108c34 2.73MB / 2.73MB  7.4s
 => => sha256:fb54daa4ae9c6c65ee26d80c5b82c6dc224058eaae7b 136.92MB / 136.92MB  430.3s
 => => sha256:b58ee5cb7152015437e4a9b3066ae9e25a26a3bef661 202.76MB / 202.76MB  534.6s
 => => sha256:1f4f297e4f699ae0f384d5cc1ea42065f58a115aa0a634 64.36MB / 64.36MB  270.9s
 => => sha256:280bbe393e788ced1dcb033580604b24de083601624337b 23.55MB / 23.55MB  87.9s
 => => sha256:1a12b4ea7c0ce04aa0e98be0a8c9942162bac71426f734 48.33MB / 48.33MB  181.5s
 => => extracting sha256:1a12b4ea7c0ce04aa0e98be0a8c9942162bac71426f734fe6d3bf98  0.6s
 => => extracting sha256:280bbe393e788ced1dcb033580604b24de083601624337be66b3ec3  0.2s
 => => extracting sha256:1f4f297e4f699ae0f384d5cc1ea42065f58a115aa0a634d427cbb18  0.9s
 => => extracting sha256:b58ee5cb7152015437e4a9b3066ae9e25a26a3bef6617d0b7f25368  2.2s
 => => extracting sha256:743f355a2ec271a051da139fb32e50f022c75db753d5108c34e51ff  0.0s
 => => extracting sha256:fb54daa4ae9c6c65ee26d80c5b82c6dc224058eaae7b92a94c911c4  1.7s
 => => extracting sha256:04c56742fba5472466f00cc7c39c3228cd6c4717e5aa9837ee95156  0.0s
 => => extracting sha256:ff68f4c56b7dc37116652db4942b512862937663c77d6a3ac4bb622  0.0s
 => [internal] load build context                                                   0.0s
 => => transferring context: 835B                                                   0.0s
 => [2/5] RUN apt-get update && apt-get install -y libssl-dev                      10.5s
 => [3/5] COPY main.cpp /app/main.cpp                                               0.0s
 => [4/5] WORKDIR /app                                                              0.0s
 => [5/5] RUN g++ main.cpp -o sha384_app -lssl -lcrypto                             0.4s
 => exporting to image                                                              0.9s
 => => exporting layers                                                             0.7s
 => => exporting manifest sha256:d45e712c2c94777bc31ca4b774928d3adfd918b6e6bfbb6    0.0s
 => => exporting config sha256:ec72d3a0f6856b481099f8c325436d741060ecc9b7397ad66    0.0s
 => => exporting attestation manifest sha256:486e37d9efec68496b114892c3f80a7ac33    0.0s
 => => exporting manifest list sha256:f8252e099388f4836d0e8c158b3eebe9f9ce0679fe    0.0s
 => => naming to docker.io/library/sha384_app:latest                                0.0s
 => => unpacking to docker.io/library/sha384_app:latest                             0.2s
[Enter your message: hello
SHA-384: 59e1748777448c69de6b800d7a33bbfb9ff1b463e44354c3553bcdb9c666fa90125a3c79f90397bdf5f6a13de828684f
```