Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Российский химико-технологический университет имени Д.И. Менделеева»

Кафедра информационных компьютерных технологий

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 4

Выполнил студент группы КС-30 Сидоров Сергей Александрович

Ссылка на репозиторий: https://github.com/MUCTR-IKT-CPP/Sidorov.S.A\_KS-30\_2sem/tree/main/lab4

Приняли: Пысин Максим Дмитриевич

Краснов Дмитрий Олегович

Дата сдачи: 07.06.2021

Оглавление

[Описание задачи. 3](#_Toc73975050)

[Описание алгоритма. 4](#_Toc73975051)

[Выполнение задачи. 5](#_Toc73975052)

[Заключение. 12](#_Toc73975053)

# Описание задачи.

В рамках лабораторной работы необходимо реализовать 1 из ниже приведенных алгоритмов хеширования:

1. MDA
2. SHA1
3. SHA2
4. Стриборг
5. RIPEMD-160

Дополнительные варианты для тех, кто хочет посложнее:

1. Luffa
2. SHA3

После завершения реализации провести следующие тесты

* сравнить результат работы созданной функции с библиотечной реализацией на 10 произвольных строк произвольной длинны, сравнение можно провести по заранее заданным строкам и заранее вычисленным хешам. В качестве задачи со звёздочкой сравнение можно производить библиотечной функцией из подключенных библиотек
* провести проверку и построить зависимости скорости расчета хеша в зависимости от размера входных данных для строк длинной (32, 64, 128, 256, 512)

# Описание алгоритма.

SHA-2 (англ. Secure Hash Algorithm Version 2 — безопасный алгоритм хеширования, версия 2) — собирательное название однонаправленных хеш-функций SHA-224, SHA-256, SHA-384 и SHA-512. Хеш-функции предназначены для создания «отпечатков» или «дайджестов» сообщений произвольной битовой длины. Применяются в различных приложениях или компонентах, связанных с защитой информации.

Хеш-функции семейства SHA-2 построены на данной основе — “Исходное сообщение после дополнения разбивается на блоки, каждый блок — на 8 слов. Алгоритм пропускает каждый блок сообщения через цикл с 64-мя или 80-ю итерациями (раундами). На каждой итерации 2 слова из восьми преобразуются, функцию преобразования задают остальные слова. Результаты обработки каждого блока складываются, сумма является значением хеш-функции.”

Алгоритм использует следующие битовые операции:

ǁ — Конкатенация,

+ — Сложение,

and — Побитовое «И»,

or — Побитовое «ИЛИ»,

xor — Исключающее «ИЛИ»,

shr — Логический сдвиг вправо,

rotr — Циклический сдвиг вправо.

В следующей таблице показаны некоторые технические характеристики различных вариантов SHA-2. «Внутреннее состояние» обозначает промежуточную хеш-сумму после обработки очередного блока данных:

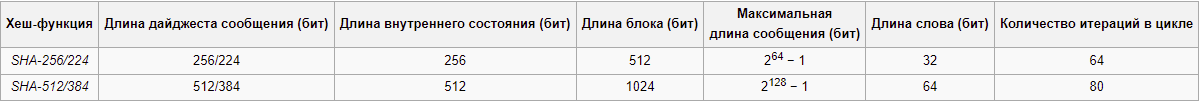


Рис. 1. Технические характеристики различных вариантов SHA-2.

# Выполнение задачи.

Реализация SHA2 (а конкретно SHA-512) была выполнена в виде класса. В классе задаются все необходимые начальные и промежуточные значения, а также методы обработки сообщения. Методом hash создаём хэш-сумму от поданной внутрь строки.

Так же было произведено сравнение хэш-суммы данной реализации и реализации на онлайн ресурсах, так как не было найдено стандартной реализации SHA-512.

1. Строка: %Ne6|]x[tp!x{&i=snJ[HssbbN}Jm9fAz6IEu&9!+(xYRBx pi

Данная реализация: c18bf112b8aa85384e7f3b3b1c55beda398a89456617b36c2af2e45e36a9aad73290a9c6cb730eb99655a01fdc588da7a7d1a7aec0ec27d8b339907399f3f2cb

Онлайн ресурс: c18bf112b8aa85384e7f3b3b1c55beda398a89456617b36c2af2e45e36a9aad73290a9c6cb730eb99655a01fdc588da7a7d1a7aec0ec27d8b339907399f3f2cb

1. Строка: fVS x$Dh8^CITLeAzRd$s[m.W0Qrxv2elGh=JGr|d0w4jV!paF

Данная реализация: c878ad60fa6e7d29d9e8a46395b47a8b6ffaf035733cce532f681e7ac56d55ce66b8bbc03c68dc80676747bde037b874a31c6e4b010a9dff80ed13345db365f2

Онлайн ресурс: c878ad60fa6e7d29d9e8a46395b47a8b6ffaf035733cce532f681e7ac56d55ce66b8bbc03c68dc80676747bde037b874a31c6e4b010a9dff80ed13345db365f2

1. Строка: fjwoP:TL0KEk[:x)cPLWY.^HKf{7;uo4=rq{mnE0!a(N5uxH;s

Данная реализация: d523afb85c545b5ee4644a7a7b16a7655347c9919ba7d04019487c482a75aca218d35f3b27ea3889a83dcbd8fe4b5e689544ad572d58a2ba6c052ba2d7eca7a3

Онлайн ресурс: d523afb85c545b5ee4644a7a7b16a7655347c9919ba7d04019487c482a75aca218d35f3b27ea3889a83dcbd8fe4b5e689544ad572d58a2ba6c052ba2d7eca7a3

1. Строка: 6PyG,.9gB7&=X5b9GMsL:H@YgbXec2U#B:{x- 7(.{bb+x,4Xm

Данная реализация: fb24a4672fb904add7d778035f6a216375ea3442edc937ebfca96327f4cbe04a57fe752ef22c62bdc7d7505d4412634c7a2e1d9bad1646106efb15367fce9530

Онлайн ресурс: fb24a4672fb904add7d778035f6a216375ea3442edc937ebfca96327f4cbe04a57fe752ef22c62bdc7d7505d4412634c7a2e1d9bad1646106efb15367fce9530

1. Строка: |,:LzAr&h7AG;{;b e#BQ#RV7v5L@{U%h:5IS6oq{Di^K }Xt8

Данная реализация: f5155119b2df929602f47e7e2541da944e328d02ce416a02de77975b6d81897efaf18febbf47bf264819b5af8dbd9a2b9b58595f420226c630abf82d5de2d062

Онлайн ресурс: f5155119b2df929602f47e7e2541da944e328d02ce416a02de77975b6d81897efaf18febbf47bf264819b5af8dbd9a2b9b58595f420226c630abf82d5de2d062

1. Строка: ONl37p,+ja,P-JK[NACtg=)kb4swSE&D7w 3YNoA,-[kP6tv%K

Данная реализация: 1a617773cf861d6119925251114fb11753e7b15eb366c84182bcb9c32618e14d4178994e1cfea0abb640e621dd8007742fd9265acd57cb1583154866548dc8b9

Онлайн ресурс: 1a617773cf861d6119925251114fb11753e7b15eb366c84182bcb9c32618e14d4178994e1cfea0abb640e621dd8007742fd9265acd57cb1583154866548dc8b9

1. Строка: cFWG$iem7ycwtb9yN{u9f93kATmr8Rg=z4lQdaL|]L]#5;$\*j=

Данная реализация: 9db2361c827fbe8e1d90e0e6b464b6a340810a6e06ba6c7d17c6a283754bc78ee9ba8dae5668cb85a773979275764e91e8f9da0a067df7049a49865fd05a0db1

Онлайн ресурс: 9db2361c827fbe8e1d90e0e6b464b6a340810a6e06ba6c7d17c6a283754bc78ee9ba8dae5668cb85a773979275764e91e8f9da0a067df7049a49865fd05a0db1

1. Строка: #XaP|$hJB2$P:=V+L]ZTw3vpmjH3fU&Vm10&|Ut9=$T$|[wgv=

Данная реализация: 3bff71c4343bf8efeb908b086ccbc5567dc4ffdf44efe0f76540c635406ed0cff4a731affc4cac15ed61c5f970f118723b087d6308b289ab48a14178cdd1745a

Онлайн ресурс: 3bff71c4343bf8efeb908b086ccbc5567dc4ffdf44efe0f76540c635406ed0cff4a731affc4cac15ed61c5f970f118723b087d6308b289ab48a14178cdd1745a

1. Строка: ;fj$Sldfc[D PxzjKgD^g;QEA5&qGF{A3,nR gq^%]s6y[-20M

Данная реализация: aa97c81a64cbd1e7a035df600837f109140163ef7c683ba0492de018a9c2ee4205b286ef96e219e8d98da58291abfd2dcc86a1e1a1a281e2d13911aaa137d924

Онлайн ресурс: aa97c81a64cbd1e7a035df600837f109140163ef7c683ba0492de018a9c2ee4205b286ef96e219e8d98da58291abfd2dcc86a1e1a1a281e2d13911aaa137d924

1. Строка: 0T]Gk4GxsCCt83${#,7K{qTyF@VR}^+M}jeB 7{#M8f:#Xd:=d

Данная реализация: 149e9d3c2d7f74102c00a01a46e8710122ed7cec4328f95d4d1afe13b3fe8188f350ec919c5f3ec7e7f61ae43894fcb16c7cef5cd087166fe0814becfde45f9d

Онлайн ресурс: 149e9d3c2d7f74102c00a01a46e8710122ed7cec4328f95d4d1afe13b3fe8188f350ec919c5f3ec7e7f61ae43894fcb16c7cef5cd087166fe0814becfde45f9d

Прилагаю скриншоты кода:

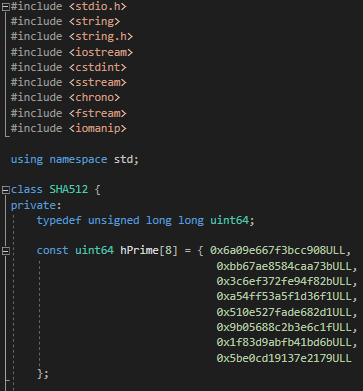


Рис. 2. Скриншот кода 1.

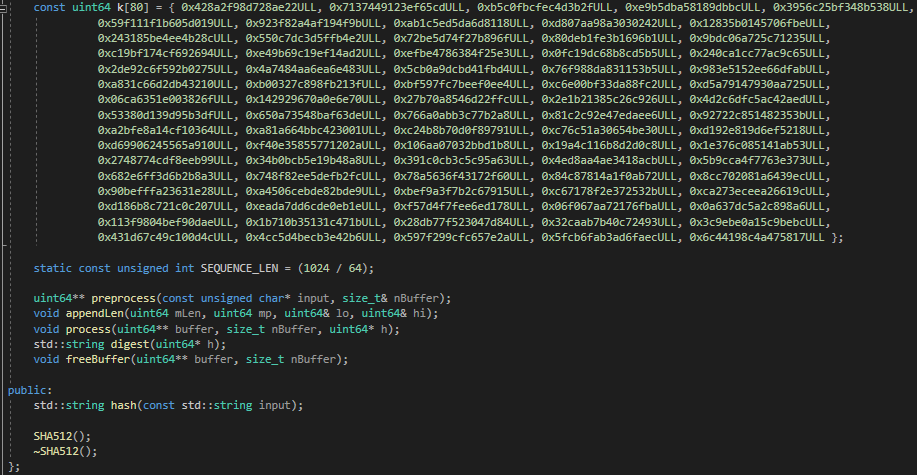


Рис. 3. Скриншот кода 2.

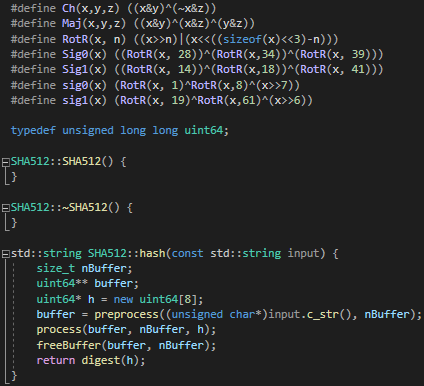


Рис. 4. Скриншот кода 3.

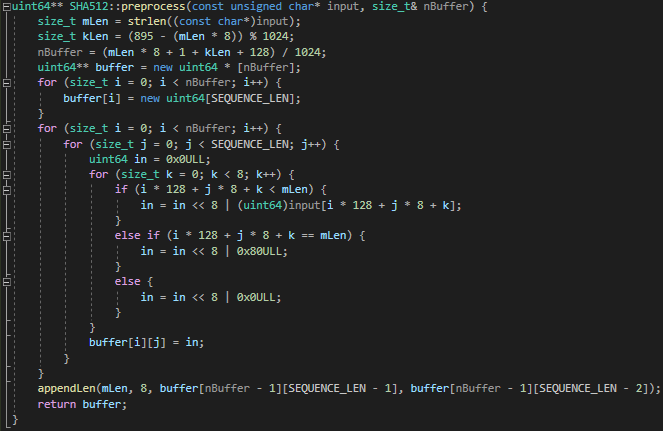


Рис. 5. Скриншот кода 4.

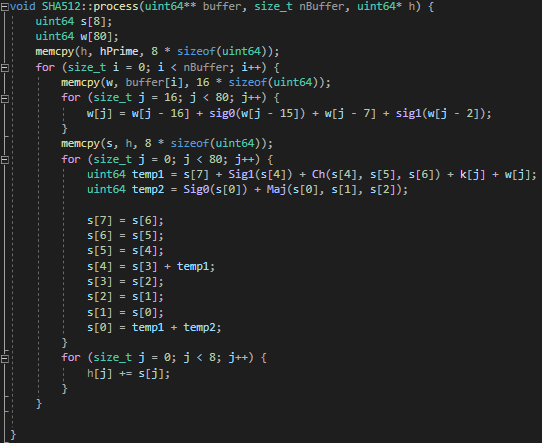


Рис. 6. Скриншот кода 5.

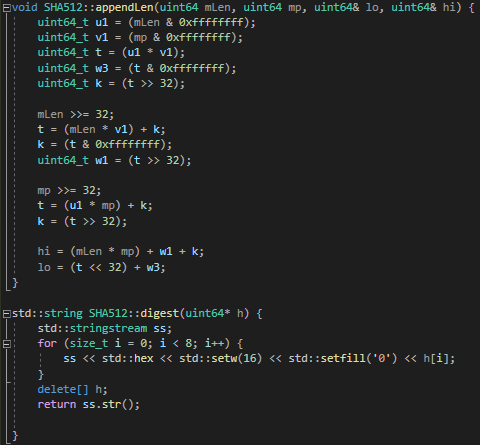


Рис. 7. Скриншот кода 6.



Рис. 8. Скриншот кода 7.

Так же был построен график для выяснения зависимости скорости расчета хэша в зависимости от размера входных данных для строк длинной (32, 64, 128, 256, 512):

# Заключение.

В ходе данной лабораторной работы была реализована функция хэширования алгоритмом SHA-2 (SHA-512). Была изучена работа самого алгоритма и написана её реализация, а также были проведены тесты корректности и производительности реализации. По графику можно сделать вывод, что при увеличении количества элементов, увеличивается и время на для расчёта хэш-суммы.