МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПЕУБЛИКИ БЕЛАРУСЬ

Учреждение образования   
«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет информационных технологий   
Кафедра информационных систем и технологий

Утверждаю

Заведующий кафедрой

\_\_\_\_\_\_\_\_\_\_\_\_ В.В. Смелов

подпись инициалы и фамилия

“\_\_\_”\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2023 г.

**ЗАДАНИЕ**

**к курсовому проектированию по дисциплине**"Защита информации и надежность информационных систем"

Специальность 1-40 05 01 03 «Информационные системы и технологии (издательско-полиграфический комплекс)»

Группа: 1

Студент: Шкабров Данила Сергеевич

**Тема:** Реализация и исследование алгоритма хэширования Keccak

**1. Срок сдачи студентом законченной работы**: «10» мая 2023г.

**2. Исходные данные к проекту:**

**2.1**. Функционально должны быть выполнены следующие задачи:

* Реализовать алгоритма хэширования Keccak на основе логических операций и циклических сдвигов.
* Разработать программное средство реализации данного метода или алгоритма.
* Составить руководство пользователя.

**2.2. Требования:**

* Необходимо провести аналитический обзор литературы по теме проекта
* Необходимо описать сферу применимости метода
* Программное средство может быть разработано на любом языке
* Архитектура приложения выбирается разработчиком
* Листинги проекта должны содержать комментарии

**3. Содержание расчетно-пояснительной записки:**

* Введение
* Постановка задачи
* Описание метода
* Описание программного средства
* Тестирование программного средства
* Руководство пользователя
* Заключение
* Список используемых источников
* Приложения

**4. Форма представления выполненного курсового проекта:**

* + Теоретическая часть курсового проекта должна быть представлена в формате MS Word.
  + Оформление записки должно быть согласно правилам.
  + Необходимые схемы, диаграммы и рисунки допускается делать в MS Office Visio или копии экрана (интерфейс).
  + Полные листинги проекта представляются в приложении.
  + К записке необходимо приложить CD (DVD), который должен содержать: пояснительную записку, листинги и файлы базы данных.

#### Календарный план

|  |  |  |  |
| --- | --- | --- | --- |
| №  п/п | Наименование этапов курсового проекта | Срок выполнения этапов проекта | Примечание |
| 1 | Введение | 24.02.2023 |  |
| 2 | Аналитический обзор литературы по теме проекта | 10.03.2023 |  |
| 3 | Разработка метода | 24.03.2023 |  |
| 4 | Разработка прототипа программного средства | 07.04.2023 |  |
| 5 | Разработка программного средства | 14.04.2023 |  |
| 6 | Тестирование программного средства | 28.04.2023 |  |
| 7 | Написание руководства пользователя | 02.05.2023 |  |
| 8 | Оформление пояснительной записки | 05.05.2023 |  |
| 9 | Сдача проекта | 10.05.2023 |  |

**5. Дата выдачи задания** «17» февраля 2023г.

Руководитель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ *Д. В. Сазонова*

(подпись)

Задание принял к исполнению \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(дата и подпись студента)

Введение

Под криптографией понимается способ передачи или хранения информации с учётом сохранения в тайне самого факта такой передачи (хранения). В отличие от криптографии, хэширование осуществляет преобразование массива входных данных произвольной длины в выходную битовую строку установленной длины, выполняемое определённым алгоритмом. Как правило, сообщение будет выглядеть как что-либо иное.

Криптографическая хеш-функция - это математический алгоритм, который отображает данные произвольного размера в битовый массив фиксированного размера.

Результат, производимый хеш-функцией, называется «хеш-суммой» или же просто «хешем», а входные данные часто называют «сообщением».

Такой пример не часто встречается в реальной работе, но он наглядно показывает, насколько хэш-функция может облегчить работу с большими объемами информации.

Например, в массив из нескольких миллионов разных строк длиной 1 млн символов нужно добавить еще одну, при условии, что там ее еще нет. Чтобы не заниматься посимвольным сравнением каждой строки, можно предварительно вычислить хэш каждой из них, и уже сделать сравнение по нему. Вся работа упрощается и ускоряется в разы.

На основе поставленной цели были выделены следующие задачи:

* Рассмотрение хэшей разной длины;
* Проведение анализа между ними;
* Вывод результата.

1. Аналитический обзор литературы

Аналоги

MD5

MD5 — предназначен для создания контрольных сумм или «отпечатков» сообщения произвольной длины и последующей проверки их подлинности. Алгоритм MD5 основан на алгоритме MD4.

Хэш-функция MD5 генерирует 128-битное хэш-значение. Изначально она была разработана для использования в криптографии, однако со временем в ней были обнаружены уязвимости, вследствие чего для этой цели она больше не подходит. И тем не менее, она по-прежнему используется для разбиения базы данных и вычисления контрольных сумм для проверки передачи файлов.

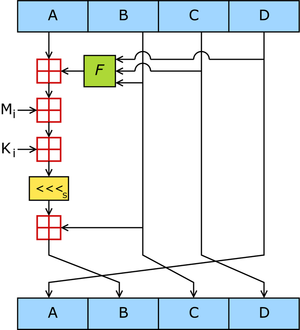


Рис. 1.1 Алгоритм работы MD5

**SHA-1**

SHA расшифровывается как Secure Hash Algorithm. SHA-1 – это первая версия алгоритма, за которой в дальнейшем последовала SHA-2.

В то время как MD5 генерирует 128-битный хэш, SHA-1 создает 160-битный (20 байт). Если представить это число в шестнадцатеричном формате, то это целое число длиной в 40 символов. Подобно MD5, этот алгоритм был разработан для криптографических приложений, но вскоре в нем также были найдены уязвимости. На сегодняшний день он считается более устойчивым к атакам в сравнении с MD5.

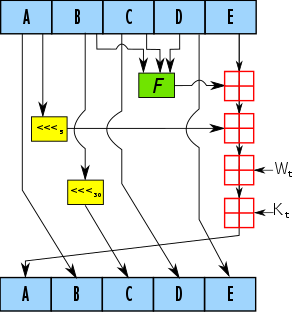


Рис. 1.2 Алгоритм работы SHA-1

**SHA-2**

Вторая версия алгоритма, SHA-2, имеет множество разновидностей. Пожалуй, наиболее часто используемая – SHA-256, которую Национальный институт стандартов и технологий (NIST) рекомендует использовать вместо MD5 и SHA-1.

Алгоритм SHA-256 возвращает 256-битное хэш-значение, что представляет собой шестнадцатеричное значение из 64 символов. Хоть это и не самый идеальный вариант, то текущие исследования показывают, что этот алгоритм значительно превосходит в безопасности MD5 и SHA-1.

Если рассматривать этот алгоритм с точки зрения производительности, то вычисление хэша с его помощью происходит на 20-30% медленнее, чем с использованием MD5 или SHA-1.

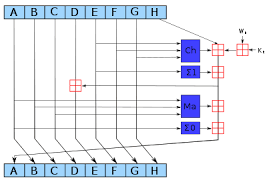


Рис. 1.3 Алгоритм работы SHA-2

2. Проектирование

В данной главе описаны технологии, которые были использованы для Реализация и исследование алгоритма хэширования Keccak. Для начала рассмотрим системы и технологии, которые мы будем использовать в данном курсовом проекте.

Windows Presentation Foundation — система для построения клиентских приложений Windows с визуально привлекательными возможностями взаимодействия с пользователем, графическая подсистема в составе .NET Framework (начиная с версии 3.0), использующая язык XAML. WPF представляет собой обширный API-интерфейс для создания настольных графических программ, имеющих насыщенный дизайн и интерактивность.

C# — это объектно- и компонентно-ориентированный язык программирования. C# предоставляет языковые конструкции для непосредственной поддержки такой концепции работы. Благодаря этому C# подходит для создания и применения программных компонентов.

Интегрированная среда разработки Visual Studio — это стартовая площадка для написания, отладки и сборки кода, а также последующей публикации приложений. Интегрированная среда разработки (IDE) представляет собой многофункциональную программу, которую можно использовать для различных аспектов разработки программного обеспечения.

1. Разработка программного средства

3.1. Создание интерфейса приложения

XAML (eXtensible Application Markup Language) - язык разметки, используемый для инициализации объектов в технологиях на платформе .NET. Применительно к WPF (а также к Silverlight) данный язык используется прежде всего для создания пользовательского интерфейса декларативным путем.

XAML – не является обязательной частью приложения, мы можем обходиться без него, создавая все элементы в файле связанного с ним кода на языке C#. Однако использование XAML все-таки несет некоторые преимущества:

* возможность отделить графический интерфейс от логики приложения, благодаря чему над разными частями приложения могут относительно автономно работать разные специалисты: над интерфейсом - дизайнеры, над кодом логики – программисты;
* компактность, понятность, простая поддержка кода.

Разработка интерфейса осуществлялась с использованием стандартных элементов, предлагаемых средой разработки Visual Studio. Конечный вариант представлен на рисунке 3.1.

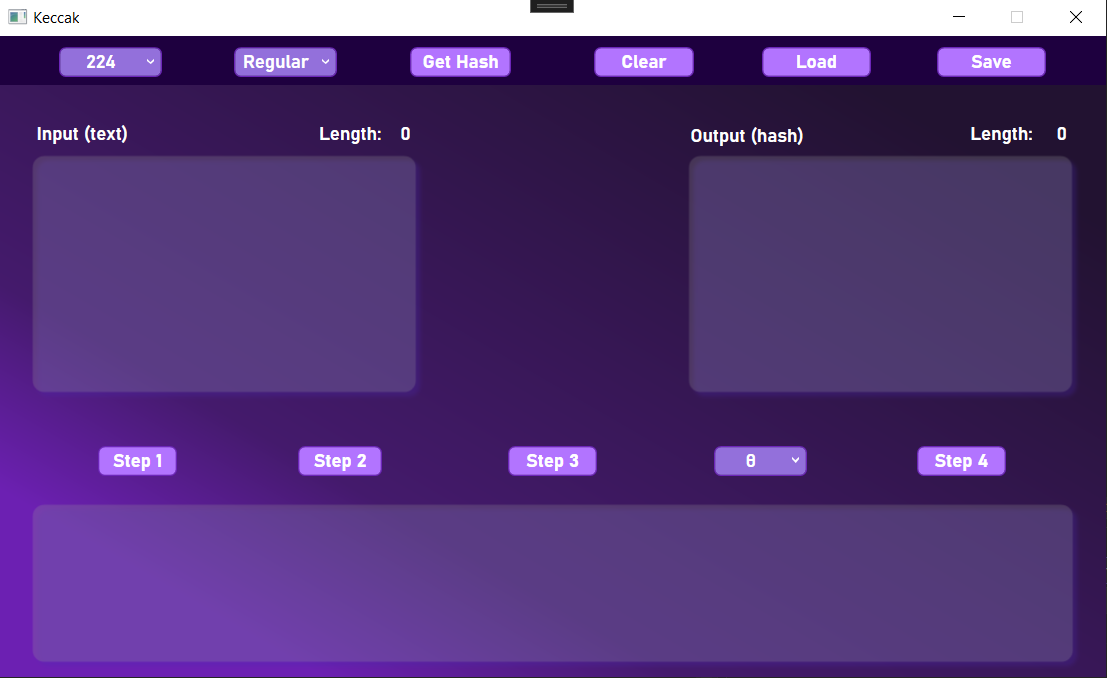


Рисунок 3.1 – Интерфейс приложения

Листинг исходного кода XAML представлен в приложении А.

* 1. Структура проекта, классы и методы

Приложение было разработано в IDE Visual Studio и представляет собой единый проект, структура которого представлена на рисунке 3.2.

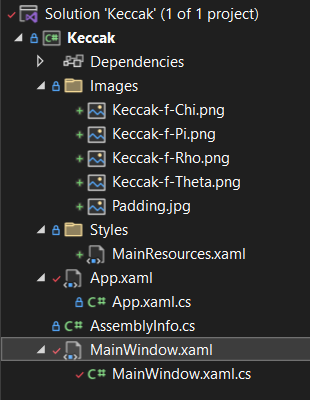


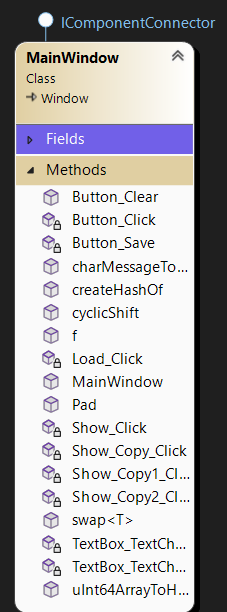
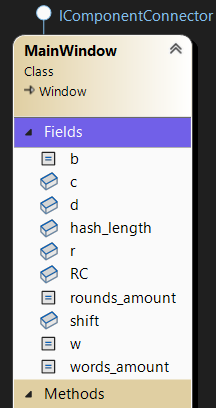
Рисунок 3.2 – Структура проекта

Самыми важными объектами в данной структуре являются объекты: MainWindow.xaml, MainWindow.xaml.cs.

MainWindow.xaml отвечает за графический интерфейс приложения. Подробно он был описан в предыдущем разделе.

MainWindow.xaml.cs является классом C#, который инициализирует программный компонент, а также содержит обработчики событий для всех UI элементов интерфейса.

Именно этот класс с помощью описанных методов может определить размер, генерировать и отображать хэш. Его структурная схема представлена на рисунке 3.3.



Рисуноки 3.3 и 3.4 – Структурная схема класса MainWindow.xaml.cs

Как можно заметить, именно этот класс с помощью описанных методов может сгенерировать хэш, можно выбрать его длину, поставить авто-получение хэша, загрузка текста с \*.txt файла и сохранение его в файл. Также можно рассмотреть поэтапно результаты алгоритмов Keccak с изображениями этапов.

Листинг кода классов представлен в приложении Б.

1. **Тестирование программного средства**

На данном этапе курсового проекта проведем тестирование разработанного приложения, чтобы удостовериться в правильности выполнения функция, а также проверить программу на отказоустойчивость.

Рассмотрим получение хэша. Кроме этого, здесь присутствует вывод количество символов сообщения, переключение между длинами хэшей, установка авто- или ручного получения хэша.

Для начала проверим работу получения хэша. Для его получения, надо установить размер хэша (рисунок 4.2), затем режим вывода (рисунок 4.3) после нажимаем на кнопку «Get Hash» (при режиме «Regular») или автоматически получим при режиме «Auto».

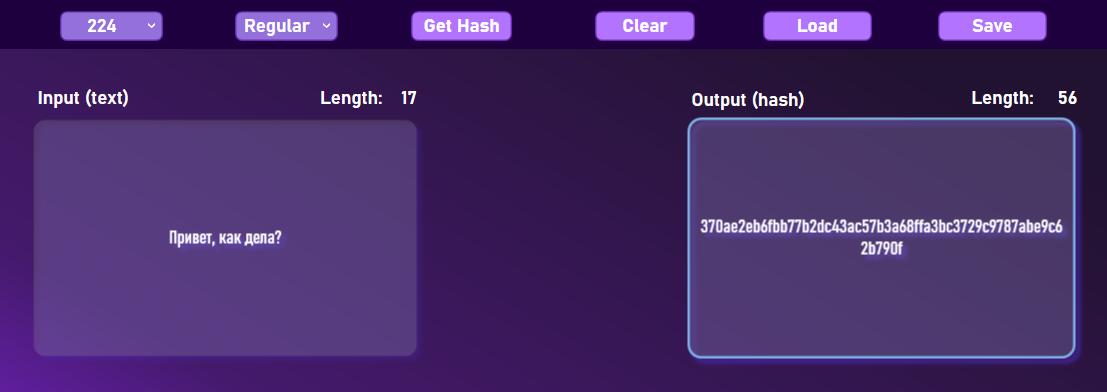
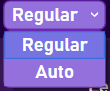
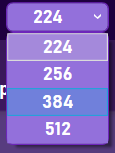


Рисунок 4.1 – Результат хэширования



Рисунки 4.2 и 4.3 – Размер и режим хэширований

Теперь рассмотрим загрузку файла для хэширования и сохранение результата в файл \*.txt. Загрузка файла в приложение на рисунках 4.4 и 4.5.

После получения хэша сохраняем хэш в файл (на рисунке 4.6) и можем увидеть результат на рисунке 4.7.

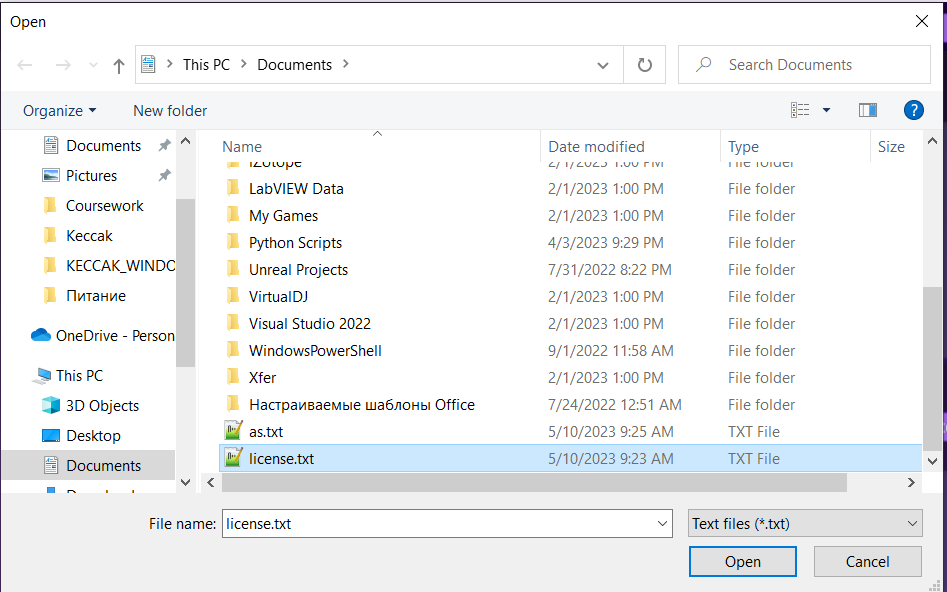


Рисунок 4.4 – Загрузка файла

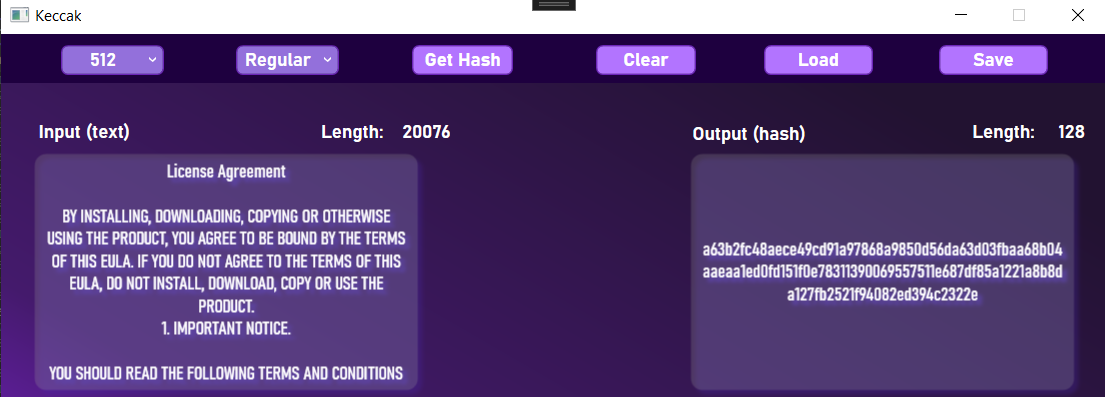


Рисунок 4.5 – Получение текста с файла и перевод его в хэш

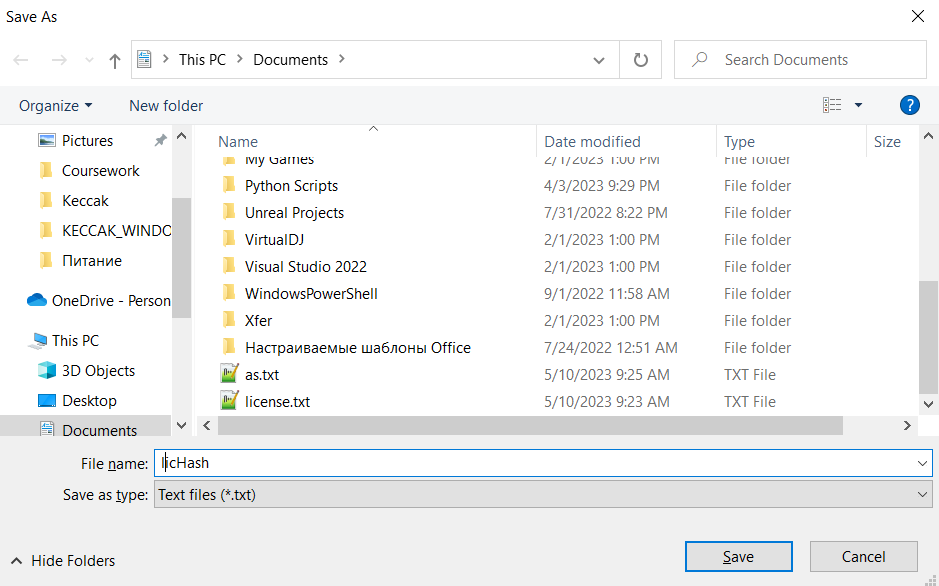


Рисунок 4.6 – Сохранение результата хэширования



Рисунок 4.7 – Файл с хэшом

Теперь перейдем к рассмотрению результатов алгоритмов хэширования с рисунками алгоритмов. Для начала нажмем на «Step 1» и увидим наши статические данные, такие как: размеры емкости и оценки, длины в байтах и т.д. (Рисунки 5.1 и 5.2)

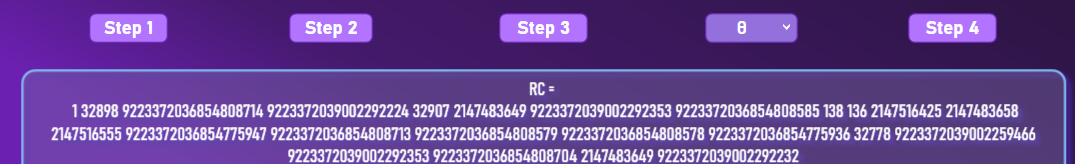


Рисунок 4.8 – Вывод статических переменных



Рисунок 4.9 – Вывод статических переменных

Далее мы можем просмотреть результаты расширения (рисунок 4.10), результат вывода 1 из алгоритмов (рисунок 4.11) и какие алгоритмы мы можем увидеть (рисунок 4.12).

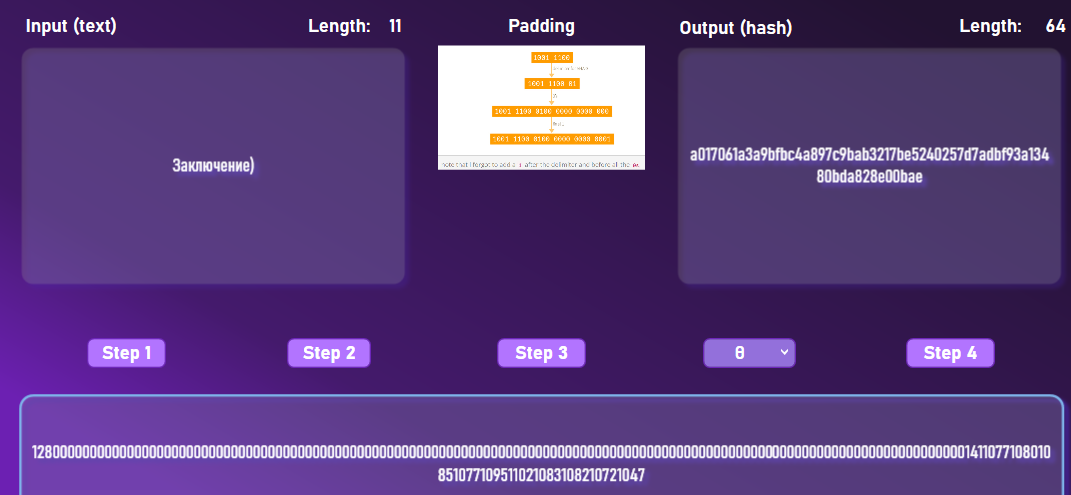


Рисунок 4.10 – Результат расширения

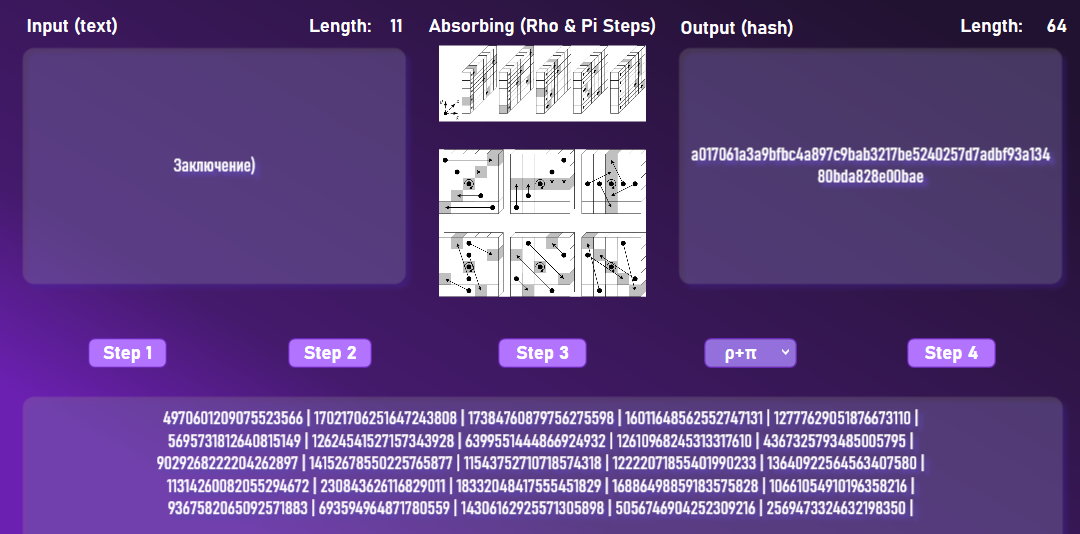


Рисунок 4.11 – Результат алгоритмов ρ и π

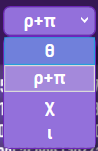


Рисунок 4.12 – Выбор алгоритма

Также есть обработка ошибок, при попытке получения результата алгоритма не используя хэширование (рисунок 4.13).

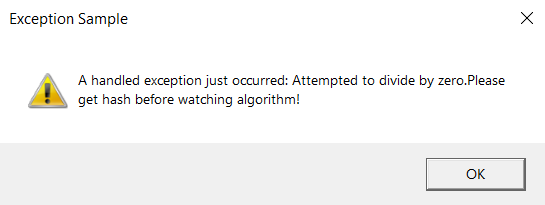


Рисунок 4.13 – Ошибка

1. **Руководство пользователя**

Основной вид приложения изображен на рисунке 5.1.

****

Рисунок 5.1 – Основной вид приложения

Для хэширования мы можем ввести сообщение или получить его с файла («Load») и нажав на «Get Hash» и получаем результат. Также в выпадающем списке можно выбрать размер хэша.

Также мы можем сохранить результат хэширования в файл нажав на («Save»)

Можно установить авто режим выбрав в выпадающем списке «Auto».

Можно очистить поля нажав на «Clear».

И последнее, для вывода результатов алгоритмов хэширования, нужно захэшировать сообщение и по шагам получать итоги.

Заключение

В ходе выполнения курсового проекта была достигнута поставленная цель по разработке приложения для получения хэшей и вывода результатов алгоритмов хэширования..

При разработке проекта выполнены следующие задачи:

* Реализовать алгоритма хэширования Keccak на основе логических операций и циклических сдвигов.
* Разработать программное средство реализации данного метода или алгоритма.
* Составить руководство пользователя.

В процессе выполнения проекта были рассмотрены различные методы хэширования. Был проведен их анализ и сравнение. В результате чего для реализации был выбран текущий метод.

На следующем этапе было спроектировано приложение, а также выбраны технологии для его реализации. Этими технологиями стали система для построения клиентских приложений WPF, язык программирования C#, а также IDE Visual Studio.

Результатом третьего этапа стало приложение, рассчитанное для использования в ОС Windows. Приложение предоставляет все регламентированные функции, а также графический интерфейс.

По результатам тестирования на следующих этапах были устранены и исправлены всевозможные ошибки, а также проверена работоспособность, и отказоустойчивость программного средства.

Было составлено руководство пользователя.

По результатам выполнения программы можно сделать вывод о том, что разработанная программа работает должным образом и требования технического задания полностью выполнены.

Список используемых источников

1. Jon's Blog [Электронный ресурс]. Режим доступа: chrome-extension://ahmkjjgdligadogjedmnogbpbcpofeeo/html/suspended.html#ttl=SHA-3%20Explained%20in%20Plain%20English%20%E2%80%93%20Jon's%20Blog&pos=3321.60009765625&uri=https://chemejon.wordpress.com/2021/12/06/sha-3-explained-in-plain-english/ . Дата доступа: 30.04.2023
2. Keccak, новый стандарт хеширования данных [Электронный ресурс]. Режим доступа: <https://habr.com/ru/articles/159073/> . Дата доступа: 30.04.2023
3. Алгоритм SHA-3 [Электронный ресурс]. Режим доступа: <https://habr.com/ru/articles/534082/> . Дата доступа: 30.04.2023

ПРИЛОЖЕНИЕ А

Файл MainWindow.xaml

|  |
| --- |
| <Window  x:Class="Keccak.MainWindow"  xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"  xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"  xmlns:d="http://schemas.microsoft.com/expression/blend/2008"  xmlns:local="clr-namespace:Keccak"  xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"  xmlns:theme="clr-namespace:Microsoft.Windows.Themes;assembly=PresentationFramework.Aero2"  Title="Keccak"  Width="900"  Height="550"  ResizeMode="CanMinimize"  mc:Ignorable="d">  <Grid>  <Grid.Background>  <LinearGradientBrush StartPoint="0.5,0" EndPoint="0.5,1">  <LinearGradientBrush.RelativeTransform>  <TransformGroup>  <ScaleTransform CenterX="0.5" CenterY="0.5" />  <SkewTransform CenterX="0.5" CenterY="0.5" />  <RotateTransform Angle="45" CenterX="0.5" CenterY="0.5" />  <TranslateTransform />  </TransformGroup>  </LinearGradientBrush.RelativeTransform>  <GradientStop Offset="1" Color="#FF6C20B2" />  <GradientStop Offset="0.772" Color="#FF441A6C" />  <GradientStop Offset="0" Color="#FF221231" />  </LinearGradientBrush>  </Grid.Background>  <Menu Margin="0,0,0,474" Background="#FF1E003F" />  <ComboBox  x:Name="ComboBoxLength"  Margin="48,9,755,0"  VerticalAlignment="Top"  HorizontalContentAlignment="Center"  Background="#FF2A1DAC"  BorderThickness="1,1,1,1"  ClipToBounds="True"  FontFamily="Bahnschrift SemiBold"  FontSize="15"  Foreground="White"  SelectedIndex="0"  Style="{DynamicResource ComboBoxStyles}">  <ComboBoxItem Content="224" />  <ComboBoxItem Content="256" />  <ComboBoxItem Content="384" />  <ComboBoxItem Content="512" />  </ComboBox>  <Button  Height="24"  Margin="329,9,476,0"  VerticalAlignment="Top"  Background="#FFB274FF"  BorderBrush="#FF6D26BA"  Click="Button\_Click"  Content="Get Hash"  FontFamily="Bahnschrift"  FontSize="15"  FontWeight="Bold"  Foreground="White"  Style="{DynamicResource ButtonStyles}" />  <TextBox  x:Name="TextBox0"  Margin="25,94,550,226"  Padding="5,5,5,5"  HorizontalContentAlignment="Center"  VerticalContentAlignment="Center"  Background="#3F9A9A9A"  BorderBrush="#00031441"  BorderThickness="2,2,2,2"  FontFamily="Bahnschrift Condensed"  FontSize="15"  FontWeight="Bold"  Foreground="White"  ScrollViewer.CanContentScroll="True"  SelectionBrush="#FF00FFED"  Style="{DynamicResource TextBoxCircleBorder}"  Text=""  TextChanged="TextBox\_TextChanged"  TextWrapping="Wrap">  <TextBox.Effect>  <DropShadowEffect Opacity="25" Color="MediumPurple" />  </TextBox.Effect>  </TextBox>  <TextBox  x:Name="TextBox1"  Margin="550,94,25,226"  Padding="5,5,5,5"  HorizontalContentAlignment="Center"  VerticalContentAlignment="Center"  AllowDrop="True"  Background="#3F9A9A9A"  BorderBrush="#00031441"  BorderThickness="2,2,2,2"  FontFamily="Bahnschrift Condensed"  FontSize="15"  FontWeight="Bold"  Foreground="White"  IsReadOnly="True"  SelectionBrush="#FF00FFED"  Style="{DynamicResource TextBoxCircleBorder}"  Text=""  TextChanged="TextBox\_TextChanged"  TextWrapping="Wrap">  <TextBox.Effect>  <DropShadowEffect Opacity="25" Color="MediumPurple" />  </TextBox.Effect>  </TextBox>  <Button  x:Name="Clear"  Height="24"  Margin="476,9,330,0"  VerticalAlignment="Top"  Background="#FFB274FF"  BorderBrush="#FF6D26BA"  Click="Button\_Clear"  Content="Clear"  FontFamily="Bahnschrift"  FontSize="15"  FontWeight="Bold"  Foreground="White"  Style="{DynamicResource ButtonStyles}" />  <ComboBox  x:Name="ComboBoxUpdate"  Margin="188,9,615,0"  VerticalAlignment="Top"  HorizontalContentAlignment="Center"  BorderThickness="1,1,1,1"  ClipToBounds="True"  FontFamily="Bahnschrift SemiBold"  FontSize="15"  Foreground="White"  SelectedIndex="0"  Style="{DynamicResource ComboBoxStyles}">  <ComboBox.Background>  <LinearGradientBrush StartPoint="0.5,0" EndPoint="0.5,1">  <GradientStop Color="Black" />  <GradientStop Offset="1" Color="White" />  </LinearGradientBrush>  </ComboBox.Background>  <ComboBoxItem Content="Regular" />  <ComboBoxItem Content="Auto" />  </ComboBox>  <Label  Margin="25,65,0,0"  HorizontalAlignment="Left"  VerticalAlignment="Top"  Content="Input (text)"  FontFamily="Bahnschrift SemiBold"  FontSize="15"  FontWeight="Bold"  Foreground="White" />  <Label  Margin="548,66,0,0"  HorizontalAlignment="Left"  VerticalAlignment="Top"  Content="Output (hash)"  FontFamily="Bahnschrift SemiBold"  FontSize="15"  FontWeight="Bold"  Foreground="White" />  <Button  x:Name="Load"  Height="24"  Margin="610,9,188,0"  VerticalAlignment="Top"  Background="#FFB274FF"  BorderBrush="#FF6D26BA"  Click="Load\_Click"  Content="Load"  FontFamily="Bahnschrift"  FontSize="15"  FontWeight="Bold"  Foreground="White"  Style="{DynamicResource ButtonStyles}" />  <Button  x:Name="Clear\_Copy1"  Height="24"  Margin="750,9,48,0"  VerticalAlignment="Top"  Background="#FFB274FF"  BorderBrush="#FF6D26BA"  Click="Button\_Save"  Content="Save"  FontFamily="Bahnschrift"  FontSize="15"  FontWeight="Bold"  Foreground="White"  Style="{DynamicResource ButtonStyles}" />  <Label  Margin="251,65,0,0"  HorizontalAlignment="Left"  VerticalAlignment="Top"  Content="Length: "  FontFamily="Bahnschrift SemiBold"  FontSize="15"  FontWeight="Bold"  Foreground="White" />  <Label  Margin="772,65,0,0"  HorizontalAlignment="Left"  VerticalAlignment="Top"  Content="Length: "  FontFamily="Bahnschrift SemiBold"  FontSize="15"  FontWeight="Bold"  Foreground="White" />  <Label  x:Name="LengthInput"  Margin="316,65,0,0"  HorizontalAlignment="Left"  VerticalAlignment="Top"  Content="0"  FontFamily="Bahnschrift SemiBold"  FontSize="15"  FontWeight="Bold"  Foreground="White" />  <Label  x:Name="LengthOutput"  Margin="841,65,0,0"  HorizontalAlignment="Left"  VerticalAlignment="Top"  Content="0"  FontFamily="Bahnschrift SemiBold"  FontSize="15"  FontWeight="Bold"  Foreground="White" />  <Button  x:Name="Show"  Height="24"  Margin="239,328,579,0"  VerticalAlignment="Top"  Background="#FFB274FF"  BorderBrush="#FF6D26BA"  Click="Show\_Click"  Content="Step 2"  FontFamily="Bahnschrift"  FontSize="15"  FontWeight="Bold"  Foreground="White"  Style="{DynamicResource ButtonStyles}" />  <Image  x:Name="Im"  Height="201"  Margin="360,94,360,0"  VerticalAlignment="Top" />  <Button  x:Name="Show\_Copy"  Height="24"  Margin="79,328,743,0"  VerticalAlignment="Top"  Background="#FFB274FF"  BorderBrush="#FF6D26BA"  Click="Show\_Copy\_Click"  Content="Step 1"  FontFamily="Bahnschrift"  FontSize="15"  FontWeight="Bold"  Foreground="White"  Style="{DynamicResource ButtonStyles}" />  <Button  x:Name="Show\_Copy1"  Height="24"  Margin="407,328,407,0"  VerticalAlignment="Top"  Background="#FFB274FF"  BorderBrush="#FF6D26BA"  Click="Show\_Copy1\_Click"  Content="Step 3"  FontFamily="Bahnschrift"  FontSize="15"  FontWeight="Bold"  Foreground="White"  Style="{DynamicResource ButtonStyles}" />  <ComboBox  x:Name="Algorithm"  Height="24"  Margin="572,328,239,0"  VerticalAlignment="Top"  HorizontalContentAlignment="Center"  Background="#FF2A1DAC"  BorderThickness="1,1,1,1"  ClipToBounds="True"  FontFamily="Bahnschrift SemiBold"  FontSize="15"  Foreground="White"  SelectedIndex="0"  Style="{DynamicResource ComboBoxStyles}">  <ComboBoxItem Content="θ" />  <ComboBoxItem Content="ρ+π" />  <ComboBoxItem Content="χ" />  <ComboBoxItem Content="ι" />  </ComboBox>  <TextBox  x:Name="Txt"  Margin="25,373,25,10"  Padding="5,5,5,5"  HorizontalContentAlignment="Center"  VerticalContentAlignment="Center"  Background="#3F9A9A9A"  BorderBrush="#00031441"  BorderThickness="2,2,2,2"  FontFamily="Bahnschrift Condensed"  FontSize="15"  FontWeight="Bold"  Foreground="White"  ScrollViewer.CanContentScroll="True"  SelectionBrush="#FF00FFED"  Style="{DynamicResource TextBoxCircleBorder}"  Text=""  TextChanged="TextBox\_TextChanged"  TextWrapping="Wrap">  <TextBox.Effect>  <DropShadowEffect Opacity="25" Color="MediumPurple" />  </TextBox.Effect>  </TextBox>  <Button  x:Name="Show\_Copy2"  Height="24"  Margin="734,328,80,0"  VerticalAlignment="Top"  Background="#FFB274FF"  BorderBrush="#FF6D26BA"  Click="Show\_Copy2\_Click"  Content="Step 4"  FontFamily="Bahnschrift"  FontSize="15"  FontWeight="Bold"  Foreground="White"  Style="{DynamicResource ButtonStyles}" />  <Image  x:Name="Im2"  Height="119"  Margin="360,177,360,0"  VerticalAlignment="Top" />  <Label  Name="Algo"  Margin="0,65,0,0"  HorizontalAlignment="Center"  VerticalAlignment="Top"  Content=""  FontFamily="Bahnschrift SemiBold"  FontSize="15"  FontWeight="Bold"  Foreground="White" VerticalContentAlignment="Center" HorizontalContentAlignment="Center" />  </Grid>  </Window> |

Файл MainResourcses.xaml

|  |
| --- |
| <ResourceDictionary  xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"  xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"  xmlns:theme="clr-namespace:Microsoft.Windows.Themes;assembly=PresentationFramework.Aero2">  <SolidColorBrush x:Key="TextBox.Static.Border" Color="#FFABAdB3" />  <SolidColorBrush x:Key="TextBox.MouseOver.Border" Color="#FF7EB4EA" />  <SolidColorBrush x:Key="TextBox.Focus.Border" Color="#FF569DE5" />  <Style x:Key="TextBoxCircleBorder" TargetType="{x:Type TextBox}">  <Setter Property="Background" Value="{DynamicResource {x:Static SystemColors.WindowBrushKey}}" />  <Setter Property="BorderBrush" Value="{StaticResource TextBox.Static.Border}" />  <Setter Property="Foreground" Value="{DynamicResource {x:Static SystemColors.ControlTextBrushKey}}" />  <Setter Property="BorderThickness" Value="1" />  <Setter Property="KeyboardNavigation.TabNavigation" Value="None" />  <Setter Property="HorizontalContentAlignment" Value="Left" />  <Setter Property="FocusVisualStyle" Value="{x:Null}" />  <Setter Property="AllowDrop" Value="true" />  <Setter Property="ScrollViewer.PanningMode" Value="VerticalFirst" />  <Setter Property="Stylus.IsFlicksEnabled" Value="False" />  <Setter Property="Template">  <Setter.Value>  <ControlTemplate TargetType="{x:Type TextBox}">  <Border  x:Name="border"  Background="{TemplateBinding Background}"  BorderBrush="{TemplateBinding BorderBrush}"  BorderThickness="{TemplateBinding BorderThickness}"  CornerRadius="10"  SnapsToDevicePixels="True">  <ScrollViewer  x:Name="PART\_ContentHost"  Focusable="false"  HorizontalScrollBarVisibility="Hidden"  VerticalScrollBarVisibility="Hidden" />  </Border>  <ControlTemplate.Triggers>  <Trigger Property="IsEnabled" Value="false">  <Setter TargetName="border" Property="Opacity" Value="0.56" />  </Trigger>  <Trigger Property="IsMouseOver" Value="true">  <Setter TargetName="border" Property="BorderBrush" Value="{StaticResource TextBox.MouseOver.Border}" />  </Trigger>  <Trigger Property="IsKeyboardFocused" Value="true">  <Setter TargetName="border" Property="BorderBrush" Value="{StaticResource TextBox.Focus.Border}" />  </Trigger>  </ControlTemplate.Triggers>  </ControlTemplate>  </Setter.Value>  </Setter>  <Style.Triggers>  <MultiTrigger>  <MultiTrigger.Conditions>  <Condition Property="IsInactiveSelectionHighlightEnabled" Value="true" />  <Condition Property="IsSelectionActive" Value="false" />  </MultiTrigger.Conditions>  <Setter Property="SelectionBrush" Value="{DynamicResource {x:Static SystemColors.InactiveSelectionHighlightBrushKey}}" />  </MultiTrigger>  </Style.Triggers>  </Style>  <Style x:Key="FocusVisual1">  <Setter Property="Control.Template">  <Setter.Value>  <ControlTemplate>  <Rectangle  Margin="2"  SnapsToDevicePixels="true"  Stroke="{DynamicResource {x:Static SystemColors.ControlTextBrushKey}}"  StrokeDashArray="1 2"  StrokeThickness="1" />  </ControlTemplate>  </Setter.Value>  </Setter>  </Style>  <SolidColorBrush x:Key="TextBox.Static.Background" Color="#FFFFFFFF" />  <Style x:Key="ComboBoxEditableTextBox" TargetType="{x:Type TextBox}">  <Setter Property="OverridesDefaultStyle" Value="true" />  <Setter Property="AllowDrop" Value="true" />  <Setter Property="MinWidth" Value="0" />  <Setter Property="MinHeight" Value="0" />  <Setter Property="FocusVisualStyle" Value="{x:Null}" />  <Setter Property="ScrollViewer.PanningMode" Value="VerticalFirst" />  <Setter Property="Stylus.IsFlicksEnabled" Value="False" />  <Setter Property="Template">  <Setter.Value>  <ControlTemplate TargetType="{x:Type TextBox}">  <ScrollViewer  x:Name="PART\_ContentHost"  Background="Transparent"  Focusable="false"  HorizontalScrollBarVisibility="Hidden"  VerticalScrollBarVisibility="Hidden" />  </ControlTemplate>  </Setter.Value>  </Setter>  </Style>  <LinearGradientBrush x:Key="ComboBox.Static.Background" StartPoint="0,0" EndPoint="0,1">  <GradientStop Offset="0.0" Color="MediumPurple" />  <GradientStop Offset="1.0" Color="MediumPurple" />  </LinearGradientBrush>  <SolidColorBrush x:Key="ComboBox.Static.Border" Color="#FF6D26BA" />  <SolidColorBrush x:Key="ComboBox.Static.Editable.Background" Color="#FFFFFFFF" />  <SolidColorBrush x:Key="ComboBox.Static.Editable.Border" Color="#FFABADB3" />  <SolidColorBrush x:Key="ComboBox.Static.Editable.Button.Background" Color="Transparent" />  <SolidColorBrush x:Key="ComboBox.Static.Editable.Button.Border" Color="Transparent" />  <LinearGradientBrush x:Key="ComboBox.MouseOver.Background" StartPoint="0,0" EndPoint="0,1">  <GradientStop Offset="0.0" Color="#FFECF4FC" />  <GradientStop Offset="1.0" Color="#FFDCECFC" />  </LinearGradientBrush>  <SolidColorBrush x:Key="ComboBox.MouseOver.Border" Color="#FF7EB4EA" />  <SolidColorBrush x:Key="ComboBox.MouseOver.Glyph" Color="#FF000000" />  <SolidColorBrush x:Key="ComboBox.MouseOver.Editable.Background" Color="#FFFFFFFF" />  <SolidColorBrush x:Key="ComboBox.MouseOver.Editable.Border" Color="#FF7EB4EA" />  <LinearGradientBrush x:Key="ComboBox.MouseOver.Editable.Button.Background" StartPoint="0,0" EndPoint="0,1">  <GradientStop Offset="0.0" Color="#FFEBF4FC" />  <GradientStop Offset="1.0" Color="#FFDCECFC" />  </LinearGradientBrush>  <SolidColorBrush x:Key="ComboBox.MouseOver.Editable.Button.Border" Color="#FF7EB4EA" />  <LinearGradientBrush x:Key="ComboBox.Pressed.Background" StartPoint="0,0" EndPoint="0,1">  <GradientStop Offset="0.0" Color="#FFDAECFC" />  <GradientStop Offset="1.0" Color="#FFC4E0FC" />  </LinearGradientBrush>  <SolidColorBrush x:Key="ComboBox.Pressed.Border" Color="#FF569DE5" />  <SolidColorBrush x:Key="ComboBox.Pressed.Glyph" Color="#FF000000" />  <SolidColorBrush x:Key="ComboBox.Pressed.Editable.Background" Color="#FFFFFFFF" />  <SolidColorBrush x:Key="ComboBox.Pressed.Editable.Border" Color="#FF569DE5" />  <LinearGradientBrush x:Key="ComboBox.Pressed.Editable.Button.Background" StartPoint="0,0" EndPoint="0,1">  <GradientStop Offset="0.0" Color="#FFDAEBFC" />  <GradientStop Offset="1.0" Color="#FFC4E0FC" />  </LinearGradientBrush>  <SolidColorBrush x:Key="ComboBox.Pressed.Editable.Button.Border" Color="#FF569DE5" />  <SolidColorBrush x:Key="ComboBox.Disabled.Background" Color="#FFF0F0F0" />  <SolidColorBrush x:Key="ComboBox.Disabled.Border" Color="#FFD9D9D9" />  <SolidColorBrush x:Key="ComboBox.Disabled.Glyph" Color="#FFBFBFBF" />  <SolidColorBrush x:Key="ComboBox.Disabled.Editable.Background" Color="#FFFFFFFF" />  <SolidColorBrush x:Key="ComboBox.Disabled.Editable.Border" Color="#FFBFBFBF" />  <SolidColorBrush x:Key="ComboBox.Disabled.Editable.Button.Background" Color="Transparent" />  <SolidColorBrush x:Key="ComboBox.Disabled.Editable.Button.Border" Color="Transparent" />  <Style x:Key="ComboBoxToggleButton" TargetType="{x:Type ToggleButton}">  <Setter Property="OverridesDefaultStyle" Value="true" />  <Setter Property="IsTabStop" Value="false" />  <Setter Property="Focusable" Value="false" />  <Setter Property="ClickMode" Value="Press" />  <Setter Property="Template">  <Setter.Value>  <ControlTemplate TargetType="{x:Type ToggleButton}">  <Border  x:Name="templateRoot"  Background="{StaticResource ComboBox.Static.Background}"  BorderBrush="{StaticResource ComboBox.Static.Border}"  BorderThickness="{TemplateBinding BorderThickness}"  CornerRadius="5"  SnapsToDevicePixels="true">  <Border  x:Name="splitBorder"  Width="{DynamicResource {x:Static SystemParameters.VerticalScrollBarWidthKey}}"  Margin="0"  HorizontalAlignment="Right"  BorderBrush="Transparent"  BorderThickness="1"  SnapsToDevicePixels="true">  <Path  x:Name="arrow"  Margin="0"  HorizontalAlignment="Center"  VerticalAlignment="Center"  Data="F1 M 0,0 L 2.667,2.66665 L 5.3334,0 L 5.3334,-1.78168 L 2.6667,0.88501 L0,-1.78168 L0,0 Z"  Fill="White" />  </Border>  </Border>  <ControlTemplate.Triggers>  <MultiDataTrigger>  <MultiDataTrigger.Conditions>  <Condition Binding="{Binding IsEditable, RelativeSource={RelativeSource AncestorType={x:Type ComboBox}}}" Value="true" />  <Condition Binding="{Binding IsMouseOver, RelativeSource={RelativeSource Self}}" Value="false" />  <Condition Binding="{Binding IsPressed, RelativeSource={RelativeSource Self}}" Value="false" />  <Condition Binding="{Binding IsEnabled, RelativeSource={RelativeSource Self}}" Value="true" />  </MultiDataTrigger.Conditions>  <Setter TargetName="templateRoot" Property="Background" Value="{StaticResource ComboBox.Static.Editable.Background}" />  <Setter TargetName="templateRoot" Property="BorderBrush" Value="{StaticResource ComboBox.Static.Editable.Border}" />  <Setter TargetName="splitBorder" Property="Background" Value="{StaticResource ComboBox.Static.Editable.Button.Background}" />  <Setter TargetName="splitBorder" Property="BorderBrush" Value="{StaticResource ComboBox.Static.Editable.Button.Border}" />  </MultiDataTrigger>  <Trigger Property="IsMouseOver" Value="true">  <Setter TargetName="arrow" Property="Fill" Value="{StaticResource ComboBox.MouseOver.Glyph}" />  </Trigger>  <MultiDataTrigger>  <MultiDataTrigger.Conditions>  <Condition Binding="{Binding IsMouseOver, RelativeSource={RelativeSource Self}}" Value="true" />  <Condition Binding="{Binding IsEditable, RelativeSource={RelativeSource AncestorType={x:Type ComboBox}}}" Value="false" />  </MultiDataTrigger.Conditions>  <Setter TargetName="templateRoot" Property="Background" Value="{StaticResource ComboBox.MouseOver.Background}" />  <Setter TargetName="templateRoot" Property="BorderBrush" Value="{StaticResource ComboBox.MouseOver.Border}" />  </MultiDataTrigger>  <MultiDataTrigger>  <MultiDataTrigger.Conditions>  <Condition Binding="{Binding IsMouseOver, RelativeSource={RelativeSource Self}}" Value="true" />  <Condition Binding="{Binding IsEditable, RelativeSource={RelativeSource AncestorType={x:Type ComboBox}}}" Value="true" />  </MultiDataTrigger.Conditions>  <Setter TargetName="templateRoot" Property="Background" Value="{StaticResource ComboBox.MouseOver.Editable.Background}" />  <Setter TargetName="templateRoot" Property="BorderBrush" Value="{StaticResource ComboBox.MouseOver.Editable.Border}" />  <Setter TargetName="splitBorder" Property="Background" Value="{StaticResource ComboBox.MouseOver.Editable.Button.Background}" />  <Setter TargetName="splitBorder" Property="BorderBrush" Value="{StaticResource ComboBox.MouseOver.Editable.Button.Border}" />  </MultiDataTrigger>  <Trigger Property="IsPressed" Value="true">  <Setter TargetName="arrow" Property="Fill" Value="{StaticResource ComboBox.Pressed.Glyph}" />  </Trigger>  <MultiDataTrigger>  <MultiDataTrigger.Conditions>  <Condition Binding="{Binding IsPressed, RelativeSource={RelativeSource Self}}" Value="true" />  <Condition Binding="{Binding IsEditable, RelativeSource={RelativeSource AncestorType={x:Type ComboBox}}}" Value="false" />  </MultiDataTrigger.Conditions>  <Setter TargetName="templateRoot" Property="Background" Value="{StaticResource ComboBox.Pressed.Background}" />  <Setter TargetName="templateRoot" Property="BorderBrush" Value="{StaticResource ComboBox.Pressed.Border}" />  </MultiDataTrigger>  <MultiDataTrigger>  <MultiDataTrigger.Conditions>  <Condition Binding="{Binding IsPressed, RelativeSource={RelativeSource Self}}" Value="true" />  <Condition Binding="{Binding IsEditable, RelativeSource={RelativeSource AncestorType={x:Type ComboBox}}}" Value="true" />  </MultiDataTrigger.Conditions>  <Setter TargetName="templateRoot" Property="Background" Value="{StaticResource ComboBox.Pressed.Editable.Background}" />  <Setter TargetName="templateRoot" Property="BorderBrush" Value="{StaticResource ComboBox.Pressed.Editable.Border}" />  <Setter TargetName="splitBorder" Property="Background" Value="{StaticResource ComboBox.Pressed.Editable.Button.Background}" />  <Setter TargetName="splitBorder" Property="BorderBrush" Value="{StaticResource ComboBox.Pressed.Editable.Button.Border}" />  </MultiDataTrigger>  <Trigger Property="IsEnabled" Value="false">  <Setter TargetName="arrow" Property="Fill" Value="{StaticResource ComboBox.Disabled.Glyph}" />  </Trigger>  <MultiDataTrigger>  <MultiDataTrigger.Conditions>  <Condition Binding="{Binding IsEnabled, RelativeSource={RelativeSource Self}}" Value="false" />  <Condition Binding="{Binding IsEditable, RelativeSource={RelativeSource AncestorType={x:Type ComboBox}}}" Value="false" />  </MultiDataTrigger.Conditions>  <Setter TargetName="templateRoot" Property="Background" Value="{StaticResource ComboBox.Disabled.Background}" />  <Setter TargetName="templateRoot" Property="BorderBrush" Value="{StaticResource ComboBox.Disabled.Border}" />  </MultiDataTrigger>  <MultiDataTrigger>  <MultiDataTrigger.Conditions>  <Condition Binding="{Binding IsEnabled, RelativeSource={RelativeSource Self}}" Value="false" />  <Condition Binding="{Binding IsEditable, RelativeSource={RelativeSource AncestorType={x:Type ComboBox}}}" Value="true" />  </MultiDataTrigger.Conditions>  <Setter TargetName="templateRoot" Property="Background" Value="{StaticResource ComboBox.Disabled.Editable.Background}" />  <Setter TargetName="templateRoot" Property="BorderBrush" Value="{StaticResource ComboBox.Disabled.Editable.Border}" />  <Setter TargetName="splitBorder" Property="Background" Value="{StaticResource ComboBox.Disabled.Editable.Button.Background}" />  <Setter TargetName="splitBorder" Property="BorderBrush" Value="{StaticResource ComboBox.Disabled.Editable.Button.Border}" />  </MultiDataTrigger>  </ControlTemplate.Triggers>  </ControlTemplate>  </Setter.Value>  </Setter>  </Style>  <ControlTemplate x:Key="ComboBoxEditableTemplate" TargetType="{x:Type ComboBox}">  <Grid x:Name="templateRoot" SnapsToDevicePixels="true">  <Grid.ColumnDefinitions>  <ColumnDefinition Width="\*" />  <ColumnDefinition Width="0" MinWidth="{DynamicResource {x:Static SystemParameters.VerticalScrollBarWidthKey}}" />  </Grid.ColumnDefinitions>  <Popup  x:Name="PART\_Popup"  Grid.ColumnSpan="2"  AllowsTransparency="true"  IsOpen="{Binding IsDropDownOpen, RelativeSource={RelativeSource TemplatedParent}}"  Placement="Bottom"  PopupAnimation="{DynamicResource {x:Static SystemParameters.ComboBoxPopupAnimationKey}}">  <theme:SystemDropShadowChrome  x:Name="shadow"  MinWidth="{Binding ActualWidth, ElementName=templateRoot}"  MaxHeight="{TemplateBinding MaxDropDownHeight}"  Color="Transparent">  <Border  x:Name="dropDownBorder"  Background="{DynamicResource {x:Static SystemColors.WindowBrushKey}}"  BorderBrush="{DynamicResource {x:Static SystemColors.WindowFrameBrushKey}}"  BorderThickness="1">  <ScrollViewer x:Name="DropDownScrollViewer">  <Grid x:Name="grid" RenderOptions.ClearTypeHint="Enabled">  <Canvas  x:Name="canvas"  Width="0"  Height="0"  HorizontalAlignment="Left"  VerticalAlignment="Top">  <Rectangle  x:Name="opaqueRect"  Width="{Binding ActualWidth, ElementName=dropDownBorder}"  Height="{Binding ActualHeight, ElementName=dropDownBorder}"  Fill="{Binding Background, ElementName=dropDownBorder}" />  </Canvas>  <ItemsPresenter  x:Name="ItemsPresenter"  KeyboardNavigation.DirectionalNavigation="Contained"  SnapsToDevicePixels="{TemplateBinding SnapsToDevicePixels}" />  </Grid>  </ScrollViewer>  </Border>  </theme:SystemDropShadowChrome>  </Popup>  <ToggleButton  x:Name="toggleButton"  Grid.ColumnSpan="2"  Background="{TemplateBinding Background}"  BorderBrush="{TemplateBinding BorderBrush}"  BorderThickness="{TemplateBinding BorderThickness}"  IsChecked="{Binding IsDropDownOpen, Mode=TwoWay, RelativeSource={RelativeSource TemplatedParent}}"  Style="{StaticResource ComboBoxToggleButton}" />  <Border  x:Name="border"  Margin="{TemplateBinding BorderThickness}"  Background="{StaticResource TextBox.Static.Background}">  <TextBox  x:Name="PART\_EditableTextBox"  Margin="{TemplateBinding Padding}"  HorizontalContentAlignment="{TemplateBinding HorizontalContentAlignment}"  VerticalContentAlignment="{TemplateBinding VerticalContentAlignment}"  IsReadOnly="{Binding IsReadOnly, RelativeSource={RelativeSource TemplatedParent}}"  Style="{StaticResource ComboBoxEditableTextBox}" />  </Border>  </Grid>  <ControlTemplate.Triggers>  <Trigger Property="IsEnabled" Value="false">  <Setter TargetName="border" Property="Opacity" Value="0.56" />  </Trigger>  <Trigger Property="IsKeyboardFocusWithin" Value="true">  <Setter Property="Foreground" Value="Black" />  </Trigger>  <Trigger SourceName="PART\_Popup" Property="HasDropShadow" Value="true">  <Setter TargetName="shadow" Property="Margin" Value="0,0,5,5" />  <Setter TargetName="shadow" Property="Color" Value="#71000000" />  </Trigger>  <Trigger Property="HasItems" Value="false">  <Setter TargetName="dropDownBorder" Property="Height" Value="95" />  </Trigger>  <MultiTrigger>  <MultiTrigger.Conditions>  <Condition Property="IsGrouping" Value="true" />  <Condition Property="VirtualizingPanel.IsVirtualizingWhenGrouping" Value="false" />  </MultiTrigger.Conditions>  <Setter Property="ScrollViewer.CanContentScroll" Value="false" />  </MultiTrigger>  <Trigger SourceName="DropDownScrollViewer" Property="ScrollViewer.CanContentScroll" Value="false">  <Setter TargetName="opaqueRect" Property="Canvas.Top" Value="{Binding VerticalOffset, ElementName=DropDownScrollViewer}" />  <Setter TargetName="opaqueRect" Property="Canvas.Left" Value="{Binding HorizontalOffset, ElementName=DropDownScrollViewer}" />  </Trigger>  </ControlTemplate.Triggers>  </ControlTemplate>  <ControlTemplate x:Key="ComboBoxTemplate" TargetType="{x:Type ComboBox}">  <Grid x:Name="templateRoot" SnapsToDevicePixels="true">  <Grid.ColumnDefinitions>  <ColumnDefinition Width="\*" />  <ColumnDefinition Width="0" MinWidth="{DynamicResource {x:Static SystemParameters.VerticalScrollBarWidthKey}}" />  </Grid.ColumnDefinitions>  <Popup  x:Name="PART\_Popup"  Grid.ColumnSpan="2"  Margin="1"  AllowsTransparency="true"  IsOpen="{Binding IsDropDownOpen, Mode=TwoWay, RelativeSource={RelativeSource TemplatedParent}}"  Placement="Bottom"  PopupAnimation="{DynamicResource {x:Static SystemParameters.ComboBoxPopupAnimationKey}}">  <theme:SystemDropShadowChrome  x:Name="shadow"  MinWidth="{Binding ActualWidth, ElementName=templateRoot}"  MaxHeight="{TemplateBinding MaxDropDownHeight}"  Color="Transparent">  <Border  x:Name="dropDownBorder"  Background="MediumPurple"  BorderBrush="#FF6D26BA"  BorderThickness="1"  CornerRadius="5">  <ScrollViewer x:Name="DropDownScrollViewer">  <Grid x:Name="grid" RenderOptions.ClearTypeHint="Enabled">  <Canvas  x:Name="canvas"  Width="0"  Height="0"  HorizontalAlignment="Left"  VerticalAlignment="Top">  <Rectangle  x:Name="opaqueRect"  Width="{Binding ActualWidth, ElementName=dropDownBorder}"  Height="{Binding ActualHeight, ElementName=dropDownBorder}"  Fill="{Binding Background, ElementName=dropDownBorder}" />  </Canvas>  <ItemsPresenter  x:Name="ItemsPresenter"  KeyboardNavigation.DirectionalNavigation="Contained"  SnapsToDevicePixels="{TemplateBinding SnapsToDevicePixels}" />  </Grid>  </ScrollViewer>  </Border>  </theme:SystemDropShadowChrome>  </Popup>  <ToggleButton  x:Name="toggleButton"  Grid.ColumnSpan="2"  Background="{TemplateBinding Background}"  BorderBrush="{TemplateBinding BorderBrush}"  BorderThickness="{TemplateBinding BorderThickness}"  IsChecked="{Binding IsDropDownOpen, Mode=TwoWay, RelativeSource={RelativeSource TemplatedParent}}"  Style="{StaticResource ComboBoxToggleButton}" />  <ContentPresenter  x:Name="contentPresenter"  Margin="{TemplateBinding Padding}"  HorizontalAlignment="{TemplateBinding HorizontalContentAlignment}"  VerticalAlignment="{TemplateBinding VerticalContentAlignment}"  Content="{TemplateBinding SelectionBoxItem}"  ContentStringFormat="{TemplateBinding SelectionBoxItemStringFormat}"  ContentTemplate="{TemplateBinding SelectionBoxItemTemplate}"  ContentTemplateSelector="{TemplateBinding ItemTemplateSelector}"  IsHitTestVisible="false"  SnapsToDevicePixels="{TemplateBinding SnapsToDevicePixels}" />  </Grid>  <ControlTemplate.Triggers>  <Trigger SourceName="PART\_Popup" Property="HasDropShadow" Value="true">  <Setter TargetName="shadow" Property="Margin" Value="0,0,5,5" />  <Setter TargetName="shadow" Property="Color" Value="#71000000" />  </Trigger>  <Trigger Property="HasItems" Value="false">  <Setter TargetName="dropDownBorder" Property="Height" Value="95" />  </Trigger>  <MultiTrigger>  <MultiTrigger.Conditions>  <Condition Property="IsGrouping" Value="true" />  <Condition Property="VirtualizingPanel.IsVirtualizingWhenGrouping" Value="false" />  </MultiTrigger.Conditions>  <Setter Property="ScrollViewer.CanContentScroll" Value="false" />  </MultiTrigger>  <Trigger SourceName="DropDownScrollViewer" Property="ScrollViewer.CanContentScroll" Value="false">  <Setter TargetName="opaqueRect" Property="Canvas.Top" Value="{Binding VerticalOffset, ElementName=DropDownScrollViewer}" />  <Setter TargetName="opaqueRect" Property="Canvas.Left" Value="{Binding HorizontalOffset, ElementName=DropDownScrollViewer}" />  </Trigger>  </ControlTemplate.Triggers>  </ControlTemplate>  <Style x:Key="ComboBoxStyles" TargetType="{x:Type ComboBox}">  <Setter Property="FocusVisualStyle" Value="{StaticResource FocusVisual1}" />  <Setter Property="Background" Value="{StaticResource ComboBox.Static.Background}" />  <Setter Property="BorderBrush" Value="{StaticResource ComboBox.Static.Border}" />  <Setter Property="Foreground" Value="{DynamicResource {x:Static SystemColors.WindowTextBrushKey}}" />  <Setter Property="BorderThickness" Value="1" />  <Setter Property="ScrollViewer.HorizontalScrollBarVisibility" Value="Auto" />  <Setter Property="ScrollViewer.VerticalScrollBarVisibility" Value="Auto" />  <Setter Property="Padding" Value="6,3,5,3" />  <Setter Property="ScrollViewer.CanContentScroll" Value="true" />  <Setter Property="ScrollViewer.PanningMode" Value="Both" />  <Setter Property="Stylus.IsFlicksEnabled" Value="False" />  <Setter Property="Template" Value="{StaticResource ComboBoxTemplate}" />  <Style.Triggers>  <Trigger Property="IsEditable" Value="true">  <Setter Property="IsTabStop" Value="false" />  <Setter Property="Padding" Value="2" />  <Setter Property="Template" Value="{StaticResource ComboBoxEditableTemplate}" />  </Trigger>  </Style.Triggers>  </Style>  <Style x:Key="FocusVisual">  <Setter Property="Control.Template">  <Setter.Value>  <ControlTemplate>  <Rectangle  Margin="2"  SnapsToDevicePixels="true"  Stroke="{DynamicResource {x:Static SystemColors.ControlTextBrushKey}}"  StrokeDashArray="1 2"  StrokeThickness="1" />  </ControlTemplate>  </Setter.Value>  </Setter>  </Style>  <SolidColorBrush x:Key="Button.Static.Background" Color="#FFDDDDDD" />  <SolidColorBrush x:Key="Button.Static.Border" Color="#FF707070" />  <SolidColorBrush x:Key="Button.MouseOver.Background" Color="#FFBEE6FD" />  <SolidColorBrush x:Key="Button.MouseOver.Border" Color="#FF3C7FB1" />  <SolidColorBrush x:Key="Button.Pressed.Background" Color="#FFC4E5F6" />  <SolidColorBrush x:Key="Button.Pressed.Border" Color="#FF2C628B" />  <SolidColorBrush x:Key="Button.Disabled.Background" Color="#FFF4F4F4" />  <SolidColorBrush x:Key="Button.Disabled.Border" Color="#FFADB2B5" />  <SolidColorBrush x:Key="Button.Disabled.Foreground" Color="#FF838383" />  <Style x:Key="ButtonStyles" TargetType="{x:Type Button}">  <Setter Property="FocusVisualStyle" Value="{StaticResource FocusVisual}" />  <Setter Property="Background" Value="{StaticResource Button.Static.Background}" />  <Setter Property="BorderBrush" Value="{StaticResource Button.Static.Border}" />  <Setter Property="Foreground" Value="{DynamicResource {x:Static SystemColors.ControlTextBrushKey}}" />  <Setter Property="BorderThickness" Value="1" />  <Setter Property="HorizontalContentAlignment" Value="Center" />  <Setter Property="VerticalContentAlignment" Value="Center" />  <Setter Property="Padding" Value="1" />  <Setter Property="Template">  <Setter.Value>  <ControlTemplate TargetType="{x:Type Button}">  <Border  x:Name="border"  Background="{TemplateBinding Background}"  BorderBrush="{TemplateBinding BorderBrush}"  BorderThickness="{TemplateBinding BorderThickness}"  CornerRadius="5"  SnapsToDevicePixels="true">  <ContentPresenter  x:Name="contentPresenter"  Margin="{TemplateBinding Padding}"  HorizontalAlignment="{TemplateBinding HorizontalContentAlignment}"  VerticalAlignment="{TemplateBinding VerticalContentAlignment}"  Focusable="False"  RecognizesAccessKey="True"  SnapsToDevicePixels="{TemplateBinding SnapsToDevicePixels}" />  </Border>  <ControlTemplate.Triggers>  <Trigger Property="IsDefaulted" Value="true">  <Setter TargetName="border" Property="BorderBrush" Value="{DynamicResource {x:Static SystemColors.HighlightBrushKey}}" />  </Trigger>  <Trigger Property="IsMouseOver" Value="true">  <Setter TargetName="border" Property="Background" Value="{StaticResource Button.MouseOver.Background}" />  <Setter TargetName="border" Property="BorderBrush" Value="{StaticResource Button.MouseOver.Border}" />  </Trigger>  <Trigger Property="IsPressed" Value="true">  <Setter TargetName="border" Property="Background" Value="{StaticResource Button.Pressed.Background}" />  <Setter TargetName="border" Property="BorderBrush" Value="{StaticResource Button.Pressed.Border}" />  </Trigger>  <Trigger Property="IsEnabled" Value="false">  <Setter TargetName="border" Property="Background" Value="{StaticResource Button.Disabled.Background}" />  <Setter TargetName="border" Property="BorderBrush" Value="{StaticResource Button.Disabled.Border}" />  <Setter TargetName="contentPresenter" Property="TextElement.Foreground" Value="{StaticResource Button.Disabled.Foreground}" />  </Trigger>  </ControlTemplate.Triggers>  </ControlTemplate>  </Setter.Value>  </Setter>  </Style>  </ResourceDictionary> |

ПРИЛОЖЕНИЕ Б

Файл Файл MainWindow.xaml.cs

|  |
| --- |
| using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  using System.Windows;  using System.Windows.Controls;  using System.Windows.Data;  using System.Windows.Documents;  using System.Windows.Input;  using System.Windows.Media;  using System.Windows.Media.Imaging;  using System.Windows.Navigation;  using System.Windows.Shapes;  using System.Diagnostics;  using Microsoft.Win32;  using System.IO;  using static System.Net.Mime.MediaTypeNames;  using System.Numerics;  namespace Keccak  {  /// <summary>  /// Interaction logic for MainWindow.xaml  /// </summary>  public partial class MainWindow : Window  {  public static void swap<T>(ref T a, ref T b)  {  T temp = a;  a = b;  b = temp;  }  public static UInt64[] RC =  {  0x0000000000000001,  0x0000000000008082,  0x800000000000808A,  0x8000000080008000,  0x000000000000808B,  0x0000000080000001,  0x8000000080008081,  0x8000000000008009,  0x000000000000008A,  0x0000000000000088,  0x0000000080008009,  0x000000008000000A,  0x000000008000808B,  0x800000000000008B,  0x8000000000008089,  0x8000000000008003,  0x8000000000008002,  0x8000000000000080,  0x000000000000800A,  0x800000008000000A,  0x8000000080008081,  0x8000000000008080,  0x0000000080000001,  0x8000000080008008  };  public static UInt16[,] shift = {  {0,36,3,41,18},  {1,44,10,45,2},  {62,6,43,15,61},  {28,55,25,21,56},  {27,20,39,8,14}  };  public const UInt16 b = 200; //state string length in bytes  public const UInt16 w = sizeof(UInt64); //word length in bytes  public const UInt16 rounds\_amount = 24;  public const UInt16 words\_amount = 5; //state string dimension size  public static UInt16 d; // hash length in bytes  public static UInt16 r; // sponge rate in bytes  public static UInt16 c; //sponge capacity in bytes  public static void Pad(Stack<char> M)  {  if (M.Count() % r == r - 1)  {  M.Push((char)0x81);  }  else  {  int zeroes\_amount = r - (M.Count() % r) - 2;  M.Push((char)0x01);  for (int i = 0; i < zeroes\_amount; i++)  {  M.Push((char)0x00);  }  M.Push((char)0x80);  }  }  public static void f(ref UInt64[,] S)  {  for (int round\_number = 0; round\_number < rounds\_amount; round\_number++)  {  UInt64[] C = { 0, 0, 0, 0, 0 };  UInt64[] D = { 0, 0, 0, 0, 0 };  UInt64[,] B = new UInt64[words\_amount, words\_amount];  for (int i = 0; i < words\_amount; ++i)  for (int j = 0; j < words\_amount; ++j)  B[i, j] = 0;  ///THETA STEP///  for (int i = 0; i < words\_amount; ++i)  C[i] = S[i, 0] ^ S[i, 1] ^ S[i, 2] ^ S[i, 3] ^ S[i, 4];  for (int i = 0; i < words\_amount; ++i)  D[i] = C[(i + 4) % words\_amount] ^ cyclicShift(C[(i + 1) % 5], 1);  for (int i = 0; i < words\_amount; ++i)  for (int j = 0; j < words\_amount; ++j)  S[i, j] = S[i, j] ^ D[i];  ///RHO & PI STEPS///  for (int i = 0; i < words\_amount; ++i)  for (int j = 0; j < words\_amount; ++j)  B[j, (2 \* i + 3 \* j) % 5] = cyclicShift(S[i, j], shift[i, j]);  ///CHI STEP///  for (int i = 0; i < words\_amount; ++i)  for (int j = 0; j < words\_amount; ++j)  S[i, j] = B[i, j] ^ ((~B[(i + 1) % 5, j]) & B[(i + 2) % 5, j]);  ///IOTA STEP///  S[0, 0] = S[0, 0] ^ RC[round\_number];  }  }  public static UInt64 cyclicShift(UInt64 word, int n)  {  n = n % (8 \* w);  return ((word << n) | (word >> (8 \* w - n)));  }  public static UInt64[][] charMessageToUInt64Words(Stack<char> M)  {  int padded\_message\_length = M.Count();  //message represented in blocks of b bytes length  UInt64[][] P = new UInt64[padded\_message\_length / r][];  for (int i = 0; i < padded\_message\_length / r; i++)  {  P[i] = new UInt64[b / w];  for (int j = 0; j < b / w; j++)  P[i][j] = 0;  for (int j = 0; j < r / w; j++)  {  //writing first r bytes to Pi block  for (int \_byte = 0; \_byte < w; \_byte++)  P[i][j] = (P[i][j] << w) | M.Pop();  }  //reversing written bytes order  Array.Reverse(P[i], 0, r / w);  }  //reversing blocks order  Array.Reverse(P);  return P;  }  public static string uInt64ArrayToHexString(UInt64[] result)  {  string Z = "";  for (int i = 0; i < result.Length; i++)  Z += BitConverter.ToString(BitConverter.GetBytes(result[i]).ToArray()).Replace("-", "");  return Z;  }  public static string createHashOf(string message, UInt16 hash\_length)  {  switch (hash\_length)  {  case 224: r = 1156; c = 448; break;  case 256: r = 1088; c = 512; break;  case 384: r = 832; c = 768; break;  case 512: r = 576; c = 1024; break;  }  d = (UInt16)(hash\_length / 8);  r /= 8;  c /= 8;  ///PADDING AND INITIALIZATION///  //writing bytes into stack because of their little-endian representation in the memory  Stack<char> M = new Stack<char>();  for (int i = 0; i < message.Length; i++)  M.Push(message[i]);  Pad(M); //padding the message to length that's multiple to r  UInt64[,] S = new UInt64[words\_amount, words\_amount]; //state string  for (int i = 0; i < words\_amount; i++)  for (int j = 0; j < words\_amount; j++)  S[i, j] = 0;  ///ABSORBING///  UInt64[][] P = charMessageToUInt64Words(M);  foreach (UInt64[] Pi in P)  {  for (int i = 0; i < words\_amount; i++)  for (int j = 0; j < words\_amount; j++)  S[i, j] ^= Pi[i + words\_amount \* j];  f(ref S);  }  ///SQUEEZING///  //this is need to be done just because otherwise it does not work properly  for (int i = 0; i < words\_amount; i++)  {  for (int j = i; j < words\_amount; j++)  {  swap<UInt64>(ref S[i, j], ref S[j, i]);  }  }  //resulting first d bytes of state string are hash value  UInt64[] result = new UInt64[d];  for (int i = 0; i < words\_amount; i++)  {  for (int j = 0; j < words\_amount; j++)  {  if (words\_amount \* i + j < result.Length)  result[words\_amount \* i + j] = S[i, j];  }  }  return uInt64ArrayToHexString(result).ToLower().Substring(0, 2 \* d);  }  public UInt16 hash\_length = 56;  public MainWindow()  {  InitializeComponent();  }  private void Button\_Click(object sender, RoutedEventArgs e)  {  hash\_length = Convert.ToUInt16(ComboBoxLength.Text);  TextBox1.Text = createHashOf(TextBox0.Text, hash\_length);  LengthOutput.Content = TextBox1.Text.Length;  }  private void TextBox\_TextChanged(object sender, TextChangedEventArgs e)  {  if (ComboBoxUpdate.Text == "Auto")  {  hash\_length = Convert.ToUInt16(ComboBoxLength.Text);  TextBox1.Text = createHashOf(TextBox0.Text, hash\_length);  }  LengthInput.Content = TextBox0.Text.Length;  LengthOutput.Content = TextBox1.Text.Length;  if (Clear.IsPressed)  {  TextBox1.Clear();  TextBox0.Clear();  }  }  private void TextBox\_TextChanged\_1(object sender, TextChangedEventArgs e)  {  LengthOutput.Content = TextBox1.Text.Length;  }  public void Button\_Clear(object sender, RoutedEventArgs e)  {  TextBox1.Clear();  TextBox0.Clear();  Txt.Clear();  Im.Source = null;  Im2.Source = null;  Algo.Content = "";  }  private void Load\_Click(object sender, RoutedEventArgs e)  {  try  {  OpenFileDialog openFileDialog = new OpenFileDialog();  openFileDialog.Filter = "Text files (\*.txt)|\*.txt";  openFileDialog.InitialDirectory = @"D:\Documents\";  if (openFileDialog.ShowDialog() == true)  {  string filename = openFileDialog.FileName;  string fileText = System.IO.File.ReadAllText(filename);  TextBox0.Text = fileText;  LengthInput.Content = TextBox0.Text.Length;  LengthOutput.Content = TextBox1.Text.Length;  MessageBox.Show("Файл открыт");  }  }  catch (Exception ex)  {  MessageBox.Show("A handled exception just occurred: " + ex.Message + "Please close file!", "Exception Sample", MessageBoxButton.OK, MessageBoxImage.Warning);  }  }  private void Button\_Save(object sender, RoutedEventArgs e)  {  try  {  SaveFileDialog saveFileDialog = new SaveFileDialog();  saveFileDialog.Filter = "Text files (\*.txt)|\*.txt";  saveFileDialog.InitialDirectory = @"D:\Documents\";  if (saveFileDialog.ShowDialog() == true)  {  string filename = saveFileDialog.FileName;  System.IO.File.WriteAllText(filename, TextBox1.Text);  MessageBox.Show("Файл сохранен");  }  }  catch (Exception ex)  {  MessageBox.Show("A handled exception just occurred: " + ex.Message + "Please close file!", "Exception Sample", MessageBoxButton.OK, MessageBoxImage.Warning);  }  }  private void Show\_Click(object sender, RoutedEventArgs e)  {  try  {  string message = TextBox0.Text;  Stack<char> M = new Stack<char>();  for (int i = 0; i < message.Length; i++)  M.Push(message[i]);  Pad(M);  Algo.Content = "Padding";  Im.Source = BitmapFrame.Create(new Uri(@"D:\BSTU\3\3-2\ISaRIS\Coursework\Keccak\Keccak\Images\Padding.jpg"));  Im2.Source = null;  string stackString = "";  foreach (int item in M)  {  stackString += item.ToString();  }  // выводим содержимое стека в TextBox  Txt.Text = stackString + "\n";  }  catch(Exception ex)  {  MessageBox.Show("A handled exception just occurred: " + ex.Message + "Please get hash before watching algorithm!", "Exception Sample", MessageBoxButton.OK, MessageBoxImage.Warning);  }  }  private void Show\_Copy\_Click(object sender, RoutedEventArgs e)  {  Algo.Content = "Data";  Im.Source = null;  Im2.Source = null;  Txt.Text = "RC = \n";  for (int i = 0; i < RC.Length; i++)  Txt.Text += " " + RC[i];  Txt.Text += "\n";  Txt.Text += "Shift = \n";  for (int i = 0; i < shift.GetLength(0); i++)  {  for (int j = 0; j < shift.GetLength(1); j++)  {  Txt.Text += shift[i, j] + " | ";  }  Txt.Text += "\n";  }  Txt.Text += "b = " + b.ToString() + " "; //state string length in bytes  Txt.Text += "w = " + w.ToString() + " "; //word length in bytes  Txt.Text += "Rounds Amount = " + rounds\_amount.ToString() + " ";  Txt.Text += "Words Amount = " + words\_amount.ToString() + " "; //state string dimension size  Txt.Text += "d = " + d.ToString() + " "; // hash length in bytes  Txt.Text += "r = " + r.ToString() + " "; // sponge rate in bytes  Txt.Text += "c = " + c.ToString(); //sponge capacity in bytes  }  private void Show\_Copy1\_Click(object sender, RoutedEventArgs e)  {  try {  string message = TextBox0.Text;  Stack<char> M = new Stack<char>();  for (int i = 0; i < message.Length; i++)  M.Push(message[i]);  Pad(M);  UInt64[,] S = new UInt64[words\_amount, words\_amount]; //state string  for (int i = 0; i < words\_amount; i++)  for (int j = 0; j < words\_amount; j++)  S[i, j] = 0;  UInt64[][] P = charMessageToUInt64Words(M);  foreach (UInt64[] Pi in P)  {  for (int i = 0; i < words\_amount; i++)  for (int j = 0; j < words\_amount; j++)  S[i, j] ^= Pi[i + words\_amount \* j];  for (int round\_number = 0; round\_number < rounds\_amount; round\_number++)  {  UInt64[] C = { 0, 0, 0, 0, 0 };  UInt64[] D = { 0, 0, 0, 0, 0 };  UInt64[,] B = new UInt64[words\_amount, words\_amount];  for (int i = 0; i < words\_amount; ++i)  for (int j = 0; j < words\_amount; ++j)  B[i, j] = 0;  if (Algorithm.Text == "θ")  {  Algo.Content = "Absorbing (Theta Step)";  Im.Source = BitmapFrame.Create(new Uri(@"D:\BSTU\3\3-2\ISaRIS\Coursework\Keccak\Keccak\Images\Keccak-f-Theta.png"));  Im2.Source = null;  Txt.Text = "";  Txt.Text += "C= ";  ///THETA STEP///  for (int i = 0; i < words\_amount; ++i)  {  C[i] = S[i, 0] ^ S[i, 1] ^ S[i, 2] ^ S[i, 3] ^ S[i, 4];  Txt.Text += C[i] + " | ";  }  Txt.Text += "\nD= ";  for (int i = 0; i < words\_amount; ++i)  {  D[i] = C[(i + 4) % words\_amount] ^ cyclicShift(C[(i + 1) % 5], 1);  Txt.Text += D[i] + " | ";  }  Txt.Text += "\nS= ";  for (int i = 0; i < words\_amount; ++i)  {  for (int j = 0; j < words\_amount; ++j)  {  S[i, j] = S[i, j] ^ D[i];  Txt.Text += S[i, j] + " | ";  }  Txt.Text += "\n";  }  ///RHO & PI STEPS///  for (int i = 0; i < words\_amount; ++i)  for (int j = 0; j < words\_amount; ++j)  B[j, (2 \* i + 3 \* j) % 5] = cyclicShift(S[i, j], shift[i, j]);  ///CHI STEP///  for (int i = 0; i < words\_amount; ++i)  for (int j = 0; j < words\_amount; ++j)  S[i, j] = B[i, j] ^ ((~B[(i + 1) % 5, j]) & B[(i + 2) % 5, j]);  ///IOTA STEP///  S[0, 0] = S[0, 0] ^ RC[round\_number];  }  if (Algorithm.Text == "ρ+π")  {  Algo.Content = "Absorbing (Rho & Pi Steps)";  Im.Source = BitmapFrame.Create(new Uri(@"D:\BSTU\3\3-2\ISaRIS\Coursework\Keccak\Keccak\Images\Keccak-f-Rho.png"));  Im2.Source = BitmapFrame.Create(new Uri(@"D:\BSTU\3\3-2\ISaRIS\Coursework\Keccak\Keccak\Images\Keccak-f-Pi.png"));  ///THETA STEP///  for (int i = 0; i < words\_amount; ++i)  C[i] = S[i, 0] ^ S[i, 1] ^ S[i, 2] ^ S[i, 3] ^ S[i, 4];  for (int i = 0; i < words\_amount; ++i)  D[i] = C[(i + 4) % words\_amount] ^ cyclicShift(C[(i + 1) % 5], 1);  for (int i = 0; i < words\_amount; ++i)  for (int j = 0; j < words\_amount; ++j)  S[i, j] = S[i, j] ^ D[i];  Txt.Text = "";  ///RHO & PI STEP///  for (int i = 0; i < words\_amount; ++i)  {  for (int j = 0; j < words\_amount; ++j)  {  B[j, (2 \* i + 3 \* j) % 5] = cyclicShift(S[i, j], shift[i, j]);  Txt.Text += B[j, (2 \* i + 3 \* j) % 5] + " | ";  }  Txt.Text += "\n";  }  ///CHI STEP///  for (int i = 0; i < words\_amount; ++i)  for (int j = 0; j < words\_amount; ++j)  S[i, j] = B[i, j] ^ ((~B[(i + 1) % 5, j]) & B[(i + 2) % 5, j]);  ///IOTA STEP///  S[0, 0] = S[0, 0] ^ RC[round\_number];  }  if (Algorithm.Text == "χ")  {  Algo.Content = "Absorbing (Chi Step)";  Im.Source = BitmapFrame.Create(new Uri(@"D:\BSTU\3\3-2\ISaRIS\Coursework\Keccak\Keccak\Images\Keccak-f-Chi.png"));  Im2.Source = null;  ///THETA STEP///  for (int i = 0; i < words\_amount; ++i)  C[i] = S[i, 0] ^ S[i, 1] ^ S[i, 2] ^ S[i, 3] ^ S[i, 4];  for (int i = 0; i < words\_amount; ++i)  D[i] = C[(i + 4) % words\_amount] ^ cyclicShift(C[(i + 1) % 5], 1);  for (int i = 0; i < words\_amount; ++i)  for (int j = 0; j < words\_amount; ++j)  S[i, j] = S[i, j] ^ D[i];  ///RHO & PI STEPS///  for (int i = 0; i < words\_amount; ++i)  for (int j = 0; j < words\_amount; ++j)  B[j, (2 \* i + 3 \* j) % 5] = cyclicShift(S[i, j], shift[i, j]);  Txt.Text = "";  ///CHI STEP///  for (int i = 0; i < words\_amount; ++i)  {  for (int j = 0; j < words\_amount; ++j)  {  S[i, j] = B[i, j] ^ ((~B[(i + 1) % 5, j]) & B[(i + 2) % 5, j]);  Txt.Text += S[i, j] + " | ";  }  Txt.Text += "\n";  }  ///IOTA STEP///  S[0, 0] = S[0, 0] ^ RC[round\_number];  }  if (Algorithm.Text == "ι")  {  Algo.Content = "Absorbing (Iota Step)";  Im.Source = null;  Im2.Source = null;  ///THETA STEP///  for (int i = 0; i < words\_amount; ++i)  C[i] = S[i, 0] ^ S[i, 1] ^ S[i, 2] ^ S[i, 3] ^ S[i, 4];  for (int i = 0; i < words\_amount; ++i)  D[i] = C[(i + 4) % words\_amount] ^ cyclicShift(C[(i + 1) % 5], 1);  for (int i = 0; i < words\_amount; ++i)  for (int j = 0; j < words\_amount; ++j)  S[i, j] = S[i, j] ^ D[i];  ///RHO & PI STEPS///  for (int i = 0; i < words\_amount; ++i)  for (int j = 0; j < words\_amount; ++j)  B[j, (2 \* i + 3 \* j) % 5] = cyclicShift(S[i, j], shift[i, j]);  ///CHI STEP///  for (int i = 0; i < words\_amount; ++i)  for (int j = 0; j < words\_amount; ++j)  S[i, j] = B[i, j] ^ ((~B[(i + 1) % 5, j]) & B[(i + 2) % 5, j]);  Txt.Text = "";  ///IOTA STEP///  S[0, 0] = S[0, 0] ^ RC[round\_number];  Txt.Text += S[0, 0];  }  }  }  }  catch (Exception ex)  {  MessageBox.Show("A handled exception just occurred: " + ex.Message + "Please get hash before watching algorithm!", "Exception Sample", MessageBoxButton.OK, MessageBoxImage.Warning);  }  }  private void Show\_Copy2\_Click(object sender, RoutedEventArgs e)  {  try {  Im.Source = null;  Im2.Source = null;  ///PADDING AND INITIALIZATION///  //writing bytes into stack because of their little-endian representation in the memory  string message = TextBox0.Text;  Stack<char> M = new Stack<char>();  for (int i = 0; i < message.Length; i++)  M.Push(message[i]);  Pad(M); //padding the message to length that's multiple to r  Algo.Content = "Squeezing";  UInt64[,] S = new UInt64[words\_amount, words\_amount]; //state string  for (int i = 0; i < words\_amount; i++)  for (int j = 0; j < words\_amount; j++)  S[i, j] = 0;  ///ABSORBING///  UInt64[][] P = charMessageToUInt64Words(M);  foreach (UInt64[] Pi in P)  {  for (int i = 0; i < words\_amount; i++)  for (int j = 0; j < words\_amount; j++)  S[i, j] ^= Pi[i + words\_amount \* j];  f(ref S);  }  ///SQUEEZING///  //this is need to be done just because otherwise it does not work properly  for (int i = 0; i < words\_amount; i++)  {  for (int j = i; j < words\_amount; j++)  {  swap<UInt64>(ref S[i, j], ref S[j, i]);  }  }  Txt.Text = "";  //resulting first d bytes of state string are hash value  UInt64[] result = new UInt64[d];  for (int i = 0; i < words\_amount; i++)  {  for (int j = 0; j < words\_amount; j++)  {  if (words\_amount \* i + j < result.Length)  result[words\_amount \* i + j] = S[i, j];  Txt.Text += result[words\_amount \* i + j] + " | ";  }  Txt.Text += "\n";  }  }  catch (Exception ex)  {  MessageBox.Show("A handled exception just occurred: " + ex.Message + "Please get hash before watching algorithm!", "Exception Sample", MessageBoxButton.OK, MessageBoxImage.Warning);  }  }  }  } |