**КОНКУРСНОЕ ЗАДАНИЕ**

***Для VII Регионального чемпионата***

 ***«Молодые профессионалы» (WORLDSKILLS RUSSIA) Санкт-Петербург 2022 года***

**компетенции**

**«РАЗРАБОТКА РЕШЕНИЙ С ИСПОЛЬЗОВАНИЕМ
БЛОКЧЕЙН ТЕХНОЛОГИЙ»**

**для возрастной категории «Юниоры»**

**14-16 лет**

*Конкурсное задание включает в себя следующие разделы:*

[1. Форма участия в конкурсе: 2](#_Toc66870131)

[2. Общее время на выполнение задания: 2](#_Toc66870132)

[3. Задание для конкурса 2](#_Toc66870133)

[4. Модули задания и необходимое время 2](#_Toc66870134)

[5. Критерии оценки. 13](#_Toc66870135)

[6. Приложения к заданию. 14](#_Toc66870136)



1. **Форма участия в конкурсе**:

Индивидуальный конкурс

1. **Общее время на выполнение задания**:

8 часов

1. **Задание для конкурса**

Целью конкурсного задания является решение кейсов, связанных с построением или поиском неисправностей в блокчейн цепи.

1. **Модули задания и необходимое время**

Таблица 1.

|  |  |  |
| --- | --- | --- |
| **Наименование модуля** | **Соревновательный день (С1, С2, С3)** | **Время на задание** |
| **A** | Основные криптографические механизмы блокчейн системы | С1 | 2 часа |
| **B** | Работа с блокчейн цепью | С1 | 2 часа |
| **C** | Проверка целостности блокчейн цепи  | С2 | 2 часа |
| D | Разработка и презентация решения | С2 | 2 часа |

Ниже представлены задачи на понимание механизмов программирования узла блокчейн. Необходимые для работы файлы, указанные в заданиях, приведены в папках с названием модулей (**МодульA**, **МодульB**, **МодульC**), расположенных на рабочем столе. Необходимые документы находятся во вложенных папках с названием соответствующем заданию.

Для представления решения необходимо создать на рабочем столе папку «МодульN». В папке «МодульN» для каждой задачи создается вложенная папка «Задача№», где N – это модуль, № - это номер задачи. В качестве решения прикладывается текстовый документ с ответами на задачу, а также проект, содержащий код решения.

**Внимание!**

При работе с файлами заданий, представленных в формате json, использовании их в качестве аргументов хэш-функций, придерживайтесь следующих правил:

- используйте только двойные кавычки «"…"» для ключей и строковых значений;

- игнорируйте символы табуляции, переноса строки, пробелы

 - {\n ‘value’: 10\n} – **неверно;**

 **-** {"value":10} – **верно.**

**Модуль A. Основные криптографические механизмы блокчейн системы Блокчейн-сеть.**

***Задание №1***

Найти хэш-значение транзакции, приведенной в формате json, используя алгоритм хэширования **sha256**. В качестве ответа укажите хэш значение в шестнадцатеричной форме. Транзакция содержится в файле **МодульA/Task1/Task1-tx.json**.

{

 "from":"0x773f8361d112a99540118a8c10242b10",

 "to":"0x383bcb1a7be9647494d87c60b075510d",

 "value":495

}

Также приведены материалы для проверки алгоритма – в файлах **Task-sample-tx.json** и **Task1-sample-tx.txt** содержится пример транзакции и ее хэш соответственно.

*InputExample*:

{
 "from":"0x93aec775d21ea6ac4385cb2db7dba415",
 "to":"0xc2297744ed8581c4309f160eaaf86542",
 "value":216
}

*OutputExample*: e3924dde5308f342f42f2a23522bad2160173dff738aeb05d7e2477a87a61d8f

***Задание №2***

Найти значение **nonce**, такое, чтобы шестнадцатеричный хэш блока **заканчивался** символами **abcd**. Блок данных представлен в файле «**МодульA/Task2/Task2\_block.json**»

Хэш считается по алгоритму **SHA3-224**. Для того чтобы правильно посчитать хэш необходимо добавить в структуру блока поле '**nonce**', которое влияет на значение хэша. Значение этого поля имеет целочисленный тип. Блок представляет собой json-объект.

В качестве ответа приведите найденный nonce, полученный хэш, заполненный json-файл.

Итоговый вид блока:

{
 "index":1046,
 "pre\_hash":"da38513f8cc1252eacbfd4117252bbd23f5114c6f388a9f55712abcd",
 "data":{
 "from":"Xaver",
 "to":"Lu",
 "value":874
 },
 "nonce":0,
 "hash":"\_\_\_\_\_"
}

***Задание №3***

Найти значение **nonce**, такое, чтобы шестнадцатеричный хэш блока **начинался двумя нулями и заканчивался тремя единицами**. Блок данных представлен в файле «**МодульA/Task3/Task3\_block.json**»

Хэш считается по алгоритму **MD5**. Для того чтобы правильно посчитать хэш необходимо добавить в структуру блока поле '**nonce**', которое влияет на значение хэша. Значение этого поля имеет целочисленный тип. Блок представляет собой json-объект.

В качестве ответа приведите найденный nonce, полученный хэш, заполненный json-файл.

Итоговый вид блока:

{
 "index":428,
 "pre\_hash":"003c827a0e1840f73ec6264f56723111",
 "data":{
 "from":"Willow",
 "to":"Bob",
 "value":1062
 },
 "nonce":0,
 "hash":"\_\_\_\_"

***Задание №4***

Известно исходное слово – «**JuniorSkills2021Final**».

Данное слово было преобразовано с помощью алгоритма хэширования **SHA256**. Затем полученный результат преобразовали в строку и прибавили к нему **шестизначное** число **X**.(конкатенация строк) После этого был сгенерирован хэш с помощью одного из алгоритмов семейства **SHA3.** Полученный хэш имеет вид 01………..10.

Если увеличить минимальный **X** на единицу, то будет получено следующее значение – «28ea4cde0cff6b66a7980653335c38de9ec0173444f7eb26c869928cbbc89e0f5bb39e5f5e4a488ea1d401c66190d951c6324edbc6f60871812ec5fa89b47bb2». (Хэш для удобства приведен в файле «Модуль1/Task4/Task4-hash»).

Участникам необходимо:

- определить значение хэша исходного слова по заданному алгоритму

- определить второй алгоритм хэширования,

- определить итоговое хэш значение,

- определить минимальное число **X**,

- определить количество различных допустимых **X**.

***Задание №5***

Была составлена строка формата WFX, где:

W - хэш значение строки «**Blockchain**», полученное по алгоритму SHA3-384 и представленное в шестнадцатеричном формате,

F - фамилия автора блокчейн платформы Ethereum\*, с большой буквы, английскими буквами,

X - натуральное шестизначное число.

После этого к слову WFX был применен один из алгоритмов семейства SHA. Полученный хэш имеет вид «cafe………..».

Если увеличить минимальный **X** на единицу, то будет получено следующее значение – «ea730f86e1512db0d7d9f7a2830a3e928ac02692883421af374412ce»., хэш хранится в файле Task5-hash.txt.

Участникам необходимо:

- определить значение хэша исходного слова по заданному алгоритму

- определить второй алгоритм хэширования,

- определить итоговое хэш значение,

- определить минимальное число **X**,

- определить другие допустимые **X**.

\*Виталик Бутерин (англ. Vitalik Buterin) – канадец российского происхождения, автор одной из крупнейших блокчейн платформ Ethereum.

**Модуль B. Работа с блокчейн цепью**

*Задание №1*

Проверить цифровую подпись блока данных с помощью алгоритма RSA.

Каталоги **Keys** и **PublicKeys** содержат в себе открытые и закрытые ключи отправителей системы. Ключ создан при помощи алгоритма RSA, длина ключа 512 бит. Содержит поля d, e, n, p, q, используемые в работе криптографической системы.

Блок представлен в виде json-объекта и записан в файле **МодульB/Task1/Task1-blok.json**, подпись блока приведена в файле **МодульB/Task1/Task1-sign.txt**.

Укажите **путь к файлу**, который вы использовали в качестве ключа для проверки подписи, а также определите используемый алгоритм хэширования, который использовался для создания подписи.

***Задание №2***

Для обеспечения целостности в блокчейн системах применяется метод построения дерева Меркла. Использование такого метода позволяет определить хэш-значение некоторого набора данных, например, транзакций в блоке. Дерево Меркла является бинарным, при его построении к каждому блоку данных применяется алгоритм хэширования, после чего, полученные значения записываются в листья дерева. Поднимаясь к корню, верви попарно объединяются, путем конкатенирования находящихся в них значений. Результатом вычислений является корень дерева – TopHash. Если на каком-то уровне дерева количество блоков данных нечетно, то крайний справа блок дублируется, добавленные хэши также считаются листьями.

Участнику необходимо:

- определить хэш первой транзакции

- определить количество листьев в дереве

- определить количество уровней в дереве, включая корень,

- вычислить значение TopHash.

Транзакции (строки), находятся в файле «**МодульB/Task2/Task2-tx.txt**». Ответом является хэш-значение. Используемый алгоритм хэширования – **SHA256**.

Для примера дается файл «**МодульB/Task2/Task2-Emaple.txt**»

TopHash для этого примера является 35209426ff916426f986fdc2ecdfcc0b91e1fc416d45ce84c38f1414c341ec17 (содержится в файле «**МодульB/Task2/Task2-Example-hash.txt**»)

***Задание №3***

Для обеспечения целостности в блокчейн системах применяется метод построения дерева Меркла. Использование такого метода позволяет определить хэш-значение некоторого набора данных, например, транзакций в блоке. Дерево Меркла является бинарным, при его построении к каждому блоку данных применяется алгоритм хэширования, после чего, полученные значения записываются в листья дерева. Поднимаясь к корню, верви попарно объединяются, путем конкатенирования находящихся в них значений. Результатом вычислений является корень дерева – TopHash. Если на каком-то уровне дерева количество блоков данных нечетно, то крайний справа блок дублируется, добавленные хэши также считаются листьями.

Участнику необходимо:

- определить хэш первой транзакции

- определить количество листьев в дереве

- определить количество уровней в дереве, включая корень,

- вычислить значение TopHash.

Транзакции (строки), находятся в файле «**МодульB/Task3/Task3-tx.txt**». Ответом является хэш-значение. Используемый алгоритм хэширования – **SHA3\_384**.

Для примера дается файл «**МодульB/Task3/Task3-Emaple.txt**»

TopHash для этого примера является 5e41bb4f15b33c352e82457ebc24ffa0add747e48e404108e48fb9a3fec1f7bfdeac069834dc02e57583f14efdaad693 (содержится в файле «**МодульB/Task3/Task3-Example-hash.txt**»)

**Модуль C. Проверка целостности блокчейн цепи**

***Задание №1***

В некоторой блокчейн системе блоки данных хранятся в текстовых файлах в json-формате и имеют структуру вида:



Такая структура представлена следующими полями:

*index*– номер блока в цепочке;

*pre\_hash* – хэш предыдущего блока;

*data*– набор транзакций в json-формате вида

*from*– идентификатор пользователя, совершающего перевод средств;

*to*– идентификатор пользователя, на чей счет переводятся средства;

*value*– объем средств;

*datahash*– хэш-значение полученное путем вычисления корня дерева Меркле от набора транзакций в блоке с помощью алгоритма **MD5**;

*creator*- идентификатор пользователя, сгенерировавшего блок;

*nonce*– поле, необходимое для вычисления хэша согласно протоколу PoW системы;

*hash*– хэш-значение от набора вышеперечисленных данных, представленных в формате json, начинается с3 нулей;

*sign*– цифровая подпись набора всех вышеперечисленных данных, учитывая хэш блока, представленных в формате json. Для подписи используется алгоритм RSA с алгоритмом хэштрования **MD5**

Папки [**Keys**](https://drive.google.com/open?id=1BlMH16Dcv1uP35IuWKM4xiMxWn2tkVEw) **и PublicKeys** содержат файлы, хранящие ключевую информацию каждого из пользователей системы – поля **d, e, n, p, q** построчно, используемые в алгоритме RSA.

Папка [**BlockChain**](https://drive.google.com/open?id=1c7wbHNd_Qai-gd6KPVLLFOsls4DeScvo) содержит несколько последовательных элементов цепочки (блоков). Некоторые файлы цепочки были преднамеренно повреждены.

**Участникам необходимо:**

Проверить элементы цепочки и обнаружить все поврежденные блоки, найти поврежденное поле и предоставить его правильное значение.

**Формат ответа:**

[номер блока] [поврежденное поле] [правильное значение]

**Пример ответа:**

5 datahash b0b5472863508f23c6fdbf8322209f1b00193b99e550f82fb5a1f6b71736b400

*Такой ответ значит, что в пятом блоке значение datahash было повреждено, и правильным значением поля является b0b5472863508f23c6fdbf8322209f1b00193b99e550f82fb5a1f6b71736b400*

***Задание №2***

Банковская система использует распределенный реестр для хранения транзакций, совершаемых пользователями.

Каждому пользователю доступны следующие операции со средствами:

-положить деньги на свой счёт [input];

-перевести деньги любому пользователю системы [send]

-получить деньги от другого пользователя [receive]);

-вывести деньги со счета в виде наличных [cash].

Так же любой пользователь может увидеть свой баланс [balance] и список совершённых транзакций[tx\_list].

Список транзакций приведен в файле **Task2-txlist.txt**.

Необходимо посчитать количество транзакций с доходами за каждое полугодие, сумму доходов в каждом полугодии, а также во сколько раз доходы пользователя, адрес которого указан в файле **Task2-adr.txt**, в первой половине года отличаются от доходов во второй половине. Результат округлить до трех знаков после запятой.

**Модуль D. Разработка и презентация решения**

Перед разработчиком стоит задача разработать архитектуру блокчейн-решения для организации работы купли-продажи живописи и фиксации авторских прав. Желательно, если вы предложите собственную идеи в любой отрасли. Ваша идея должна быть креативной, актуальной и целесообразной в сфере применения блокчейн-технологий

Необходимо разработать архитектуру решения. Указать основные функциональные модули системы и их взаимосвязь. Отразить основные механизмы формирования блокчейн-цепи (транзакции, блоки, генезисблок, уязвимости, консенсус).

Предложите эскизы графического интерфейса пользователя, используя любые средства (графические редакторы, бумага, ручка и т.д.)

Презентовать разработанную архитектуру решения.

Регламент выступления не более 7 минут.

1. **Критерии оценки.**

Таблица 2.

|  |  |
| --- | --- |
| **Критерий** | **Баллы** |
| **Судейские аспекты** | **Объективная оценка** | **Общая оценка** |
| **A** | **Основные криптографические механизмы блокчейн системы** | 1 | 27 | 28 |
| **B** | **Работа с блокчейн цепью** | 1 | 23 | 24 |
| **C** | **Проверка целостности блокчейн цепи** | 1 | 26 | 27 |
| **D** | **Разработка и презентация решения** | 14 | 7 | 21 |
| **Итого** | 17 | 83 | 100 |

1. **Приложения к заданию.**

Приложение 1. Папка «Задачи», содержащий файлы с конкурсными заданиями.