Assignment 2

1. Exploring hash

To begin, the following hash tool will be used:

https://forks.fun/#/blockchain/*hash*

This tool has a simple function that automatically runs a hash function ("SHA- 256") on any value that has been entered into the “Data” field.

As discussed in Module 2, a hash function is a special function that converts a string of any length into a fixed-length string. Hash functions have four important characteristics that make them very useful for test blockchains:

1. A hash function creates a fixed-length output, whatever the input.
   Answer **questions 1 and 2**.

2. Hash functions give an “avalanche effect”, which means that a minimal change in the input will create a completely **random** change in the output.
   Answer **questions 3 and 4**.

3. Hash functions are deterministic, i.e. the same input will always create the same output.
   Answer **question 5**.

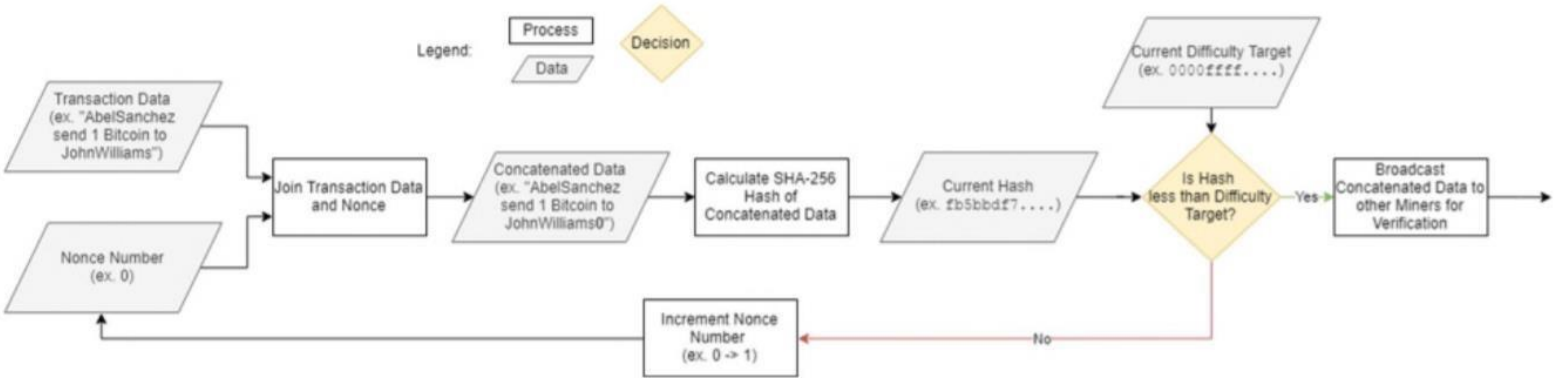
4. Hash functions are one-way functions. With the input, you can always calculate the output, but with the output, you cannot calculate what the input value was.

2. The "target difficulty " and the "nonce"

In section 2 the “target difficulty” was explained. As a reminder, the target difficulty concept is described the following way:

- The target difficulty is a value that limits the hashing process. In order to mine, it is not only necessary to find a **hash**, but that the value must be lower than the **target difficulty**. Overall, it is said that it consists of a series of initial zeros of the hash, but that is an excessive simplification.
- Finding that hash is a matter of probability, i.e., it is a trial-and-error process. That´s why Bitcoin must adjust the target every 2016 blocks, or every two weeks. If the hash rate in the network increases, the target value will decrease so that, statistically, a block is mined every ten minutes or so.

Therefore, the objective of “mining” in Bitcoin, or in another cryptocurrency, is to get data whose hash is less than a particular target difficulty. The process is described below.



As an example, you can assume that the target difficulty is represented with a hash that starts with a zero followed by 63 f[1], as shown below:

Target difficulty:

```
0fffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffff
```

"f" represents the number 15 in hexadecimal. In this case, any hash that starts with a “1” or more will be higher than the target difficulty and any hash that starts with a “0” will be lower than the target difficulty. See the following examples:

Higher:

```
1fffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffff
ffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffff
e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855
```

Lower:

```
0111fffffffffffffffffffffffffffffffffffffffffffffffffffffffffff
0ffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffe
03b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855
```

Answer **question 6**.

In the previous exercise you have seen that any value starting with “0” is lower than the target difficulty and any other value is higher. Due to there being sixteen potential options for the first character of a hash (0-9, a, b, c, d, e, or f), it can be said that, on average, one of sixteen random hash values will be lower than the target difficulty.

Remembering that the "avalanche effect" essentially creates a completely random hash by simply changing one digit of the input, it can be assumed that each time the hash value is changed, there is a one in sixteen chance that the hash will be lower than the target difficulty.

Go back to the hash tool from the last exercise. Enter your name in the field, followed by “0” (e.g., “AbelSanchez0”).

1. Like analogical test mining, the name will be the “data” of the block and the number of the “nonce”.
   Answer **question 7**.

1. Often, the hash values are represented as hexadecimal numbers, which is just a different way of representing a number. Instead of using only numbers from zero to nine to represent digits, the hexadecimal system adds “a”, “b”, “c”, “d”, “e” and “f” to represent the numbers “10”, “11”, “12”, “13”, “14” and “15” respectively. This means that each hexadecimal character can have a value from zero to fifteen.

2. Now increase the nonce between zero and sixty-four (e.g., “Abel Sanchez0”, “Abel Sanchez64”). Look out for the results.
   Answer **question 8**.

3. Now assume that a person is hired to try and find new nonce values that are lower than the target difficulty. This person can test exactly sixty-four new nonce values every minute.
   Answer **question 9**.

4. Finally, assume that the target difficulty has been changed. Instead of identifying a hash that starts with a single zero, the new difficulty hash must start with two zeros.

   Now, the probability of finding a valid nonce is much lower. With a zero, the probability of finding a valid nonce was one in sixteen. With two zeros, however, the probability of finding a valid nonce is $(1/16)^2$ or one in 256 hashes.
   Answer **question 10**.

The problems listed above demonstrate a process analogous to that which occurs in proof-of-work mining. Obviously, instead of having people physically prove that a nonce is lower than a certain value, miners use special computers (often called mining rigs). In many cases, these special computers can test more than 50 TeraHashes per second (50×10^{12}). With this magnitude of power, you would find the solution to your target difficulty in a very short time.

3. Short answer – the future of mining

In many proof-of-work systems, mining is a race against time. The first miner to find a nonce according to the target difficulty receives cryptocoins (Bitcoin miners currently receive 6.25 Bitcoins), and the race begins again. The miners are strongly encouraged to "win the race". There are two main strategies to speed up the process:

1. Buy better hardware. A high-end computer processor can process approximately 20×10^6 hashes per second, while more powerful mining rigs can process a million times faster, at a speed of over 50×10^{12} hashes per second.

2. Join a mining pool, where many users pool their computer resources to find a hash according to the target difficulty. If someone in the pool receives the award, it is shared equally, depending on how many hashes each machine has processed.

Investigate both strategies and find out if these strategies are becoming more or less attractive.

Answer **questions 11-13**.

3. Etherscan Network Hash Rate

See the hash rate chart in Etherscan:

https://etherscan.io/chart/*hash*rate

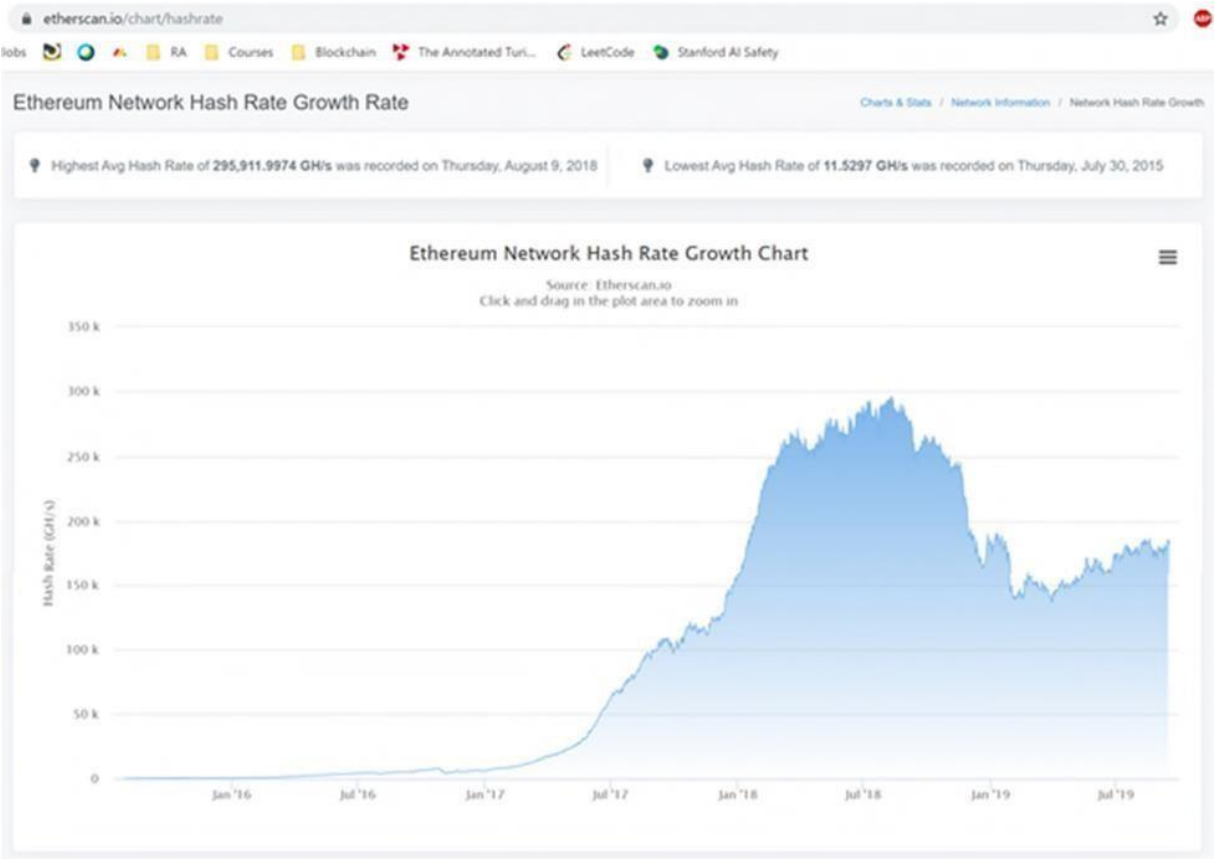


Figure 1: Average hash rate

In the previous exercise, we have seen the hash rate of a single machine, and that personal computers can process 20 x 10^6 hashes per second, the high-end mining rigs, however, process up to 50 x 10^{12} hashes per second.

Answer **question 14**.

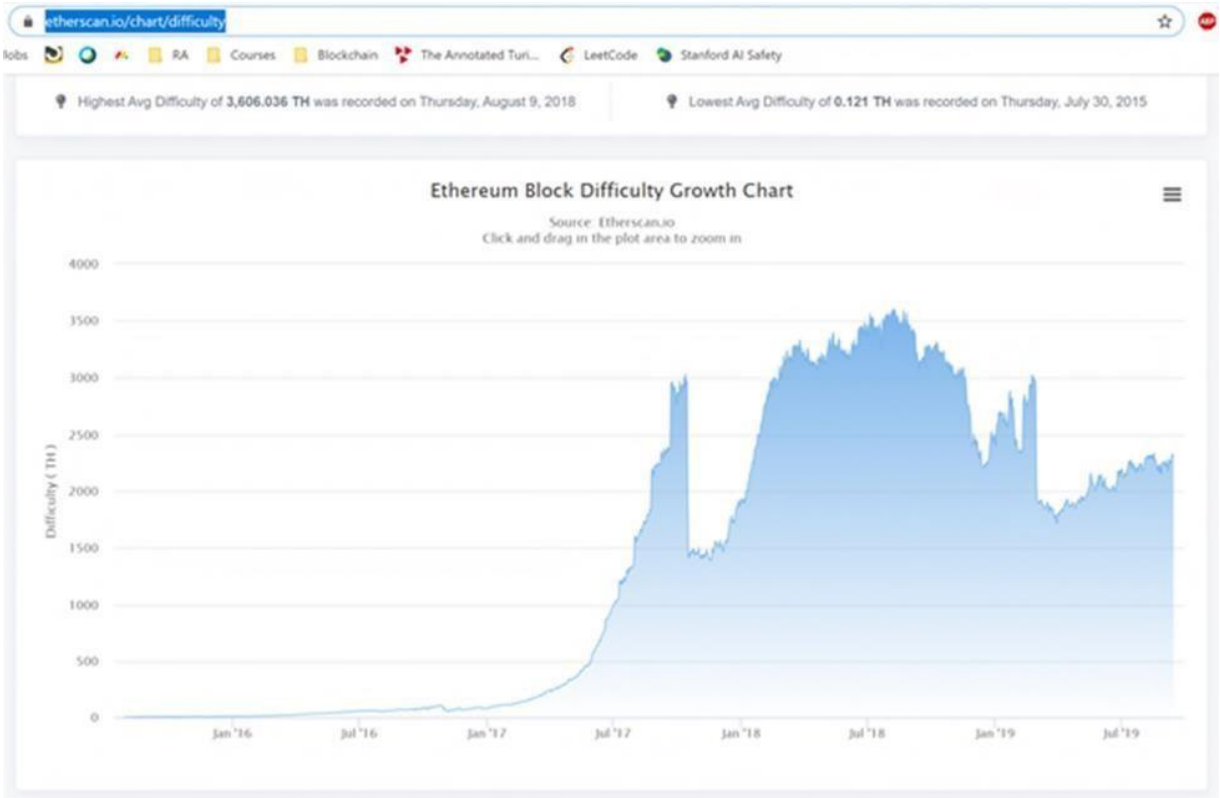
Over time, the hash rate of the Ethereum network increased steadily during the first three and a half years of the network's life. After that, it was regulated between 250k and 300k for most of 2018, and then dropped below 200k.

Answer question **15**.

Ethereum Difficulty

Finally, see the target difficulty in Ethereum over time:

https://etherscan.io/chart/difficulty



The Ethereum target difficulty operates very similarly to Bitcoin´s target difficulty and to the previous exercise on blockchain mining. Unfortunately, the hash target in Ethereum is not shown in Ethereum browsers (for more information, see: “Why are there no initial zeros in the Ethereum hash block?”).

Answer **question 16**.

4. Blocks and chains

This part of the activity is **voluntary**.

Now that you understand what the target difficulty is in identifying a valid block in proof-of-work mining, you can continue with the hashes to understand the concepts of "block" and "chain".

Block headers, as we now know, contain "data" that includes transaction data and a nonce. Strategically, block headers also contain the hash of the previous block. By using the previous hash in the new block, if a wrongdoer tries to change a blockchain transaction, it will affect the hash of that transaction and, therefore, all future transactions. For example:

> *Dr. Sanchez has created a cryptocurrency called SanchezCoin. He has sent you 1 SanchezCoin in the first block of the cryptocurrency network. The transaction is registered with multiple other transactions in that first block.*

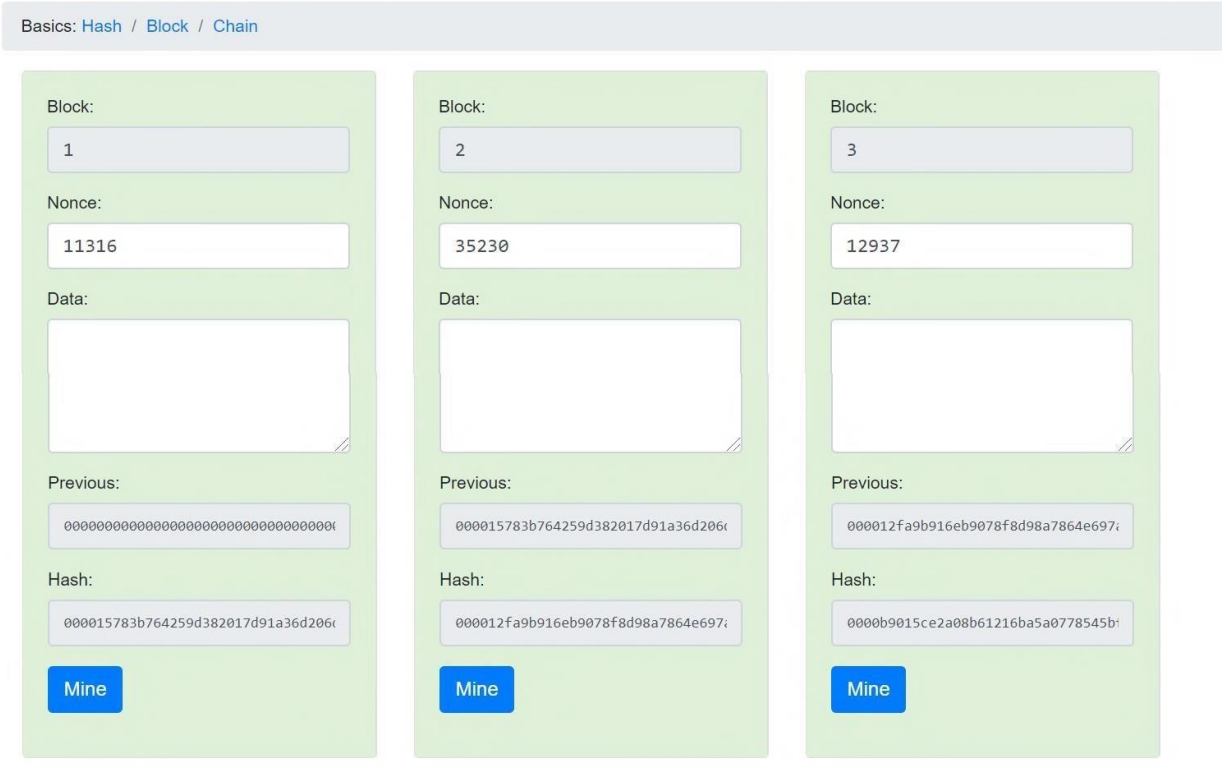
Over time, transactions between more participants are completed. When you get to the fifth block, Professor Williams offers you a deal that would be very favorable to you: he sells you his new sports car for the price of 2 SanchezCoin.

Professor Williams doesn´t know that you only have 1 SanchezCoin. You decide to lie and pretend you have 2.

For this exercise, the following blockchain tool will be used: https://forks.fun/#/blockchain/chain.

In this network, the target difficulty is at four initial zeros (0000f…) and the "Hash" is the concatenation of four fields: "Block", "Nonce", "Data" and "Previous". You can enter the default parameters into the tool in order to see that the hash value is identical:

Block:	1
Nonce:	11316
Data:	[empty]
Previous:	`0000000000000000000000000000000000000000000000000000000000000000`
Hash:	`000015783b764259d382017d91a36d206d0600e2cbb3567748f46a33fe9297cf`



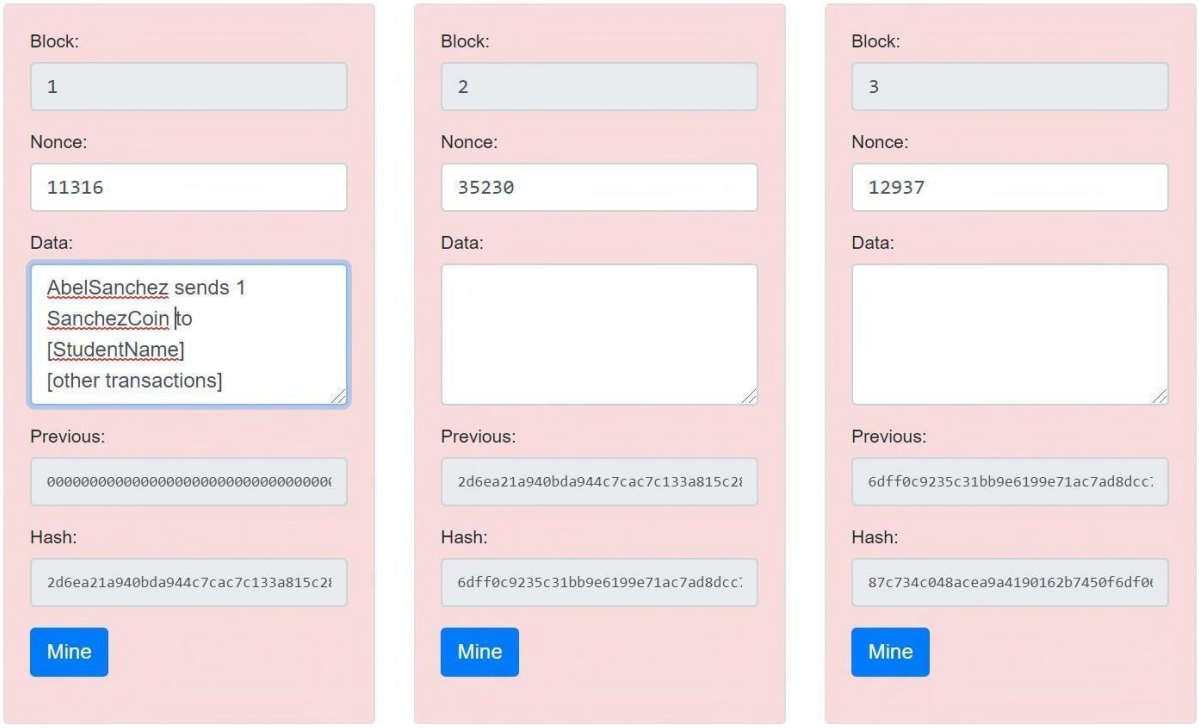
Therefore, if you concatenate the data and enter it into the hash tool this way:

1113160000000000000000000000000000000000000000000000000000000000000000

You should see the same hash (00001578...).

You will start by simulating the first four blocks of this blockchain network.

1) The first block includes SanchezCoin that Dr. Sanchez has sent you, in addition to other transactions in which you did not participate:



The tool includes a mining button. You should not increase the nonce at hand, when you click on the button, the computer will automatically process nonce values, one by one. If the resulting hash is lower than the target difficulty, it will stop.

When you click on "Mine", what nonce value did the program find?

2) The following blocks will now be mined. None of these transactions concern you, so enter [other transactions] in the "Data" field. Click on "Mine" in each new block and take note of the nonce values:

   a. **Nonce block 2:** _____

   b. **Nonce block 3:** _____

   c. **Nonce block 4:** _____

3) Finally. The time has come and you want Professor Williams to believe that you have 2 SanchezCoin. For some reason, Professor Williams doesn´t have access to the ledger itself, but he does have access to the hash values of the first four blocks. He asks you for

a copy of the ledger to verify it and you change the first transaction “AbelSanchez sends you 1 SanchezCoin” to “AbelSanchez sends you 2 SanchezCoin”.

When you send Professor Williams the first transfer and he creates a hash based on the data provided, what hash value will he calculate?

_____

Will Professor Williams believe you and hand you over the car? Why yes or why not?

_____

_____

_____