Blockchain

1- Overview

Public-key cryptography uses a public and private key to encrypt messages.

This system can be used to verify messages by creating a unique signature that can be verified by anybody.

This process is the basis for transactions on blockchain networks, where individuals exchange currency without needing to trust one another.

In this activity, you will use a widget to simulate several blockchain transactions, including mining blocks and adding them to the shared blockchain ledger.

2- Vocabulary

Match the words to their definition:
Public Key ; Ledger ; Symmetric Encryption ; Encryption ; Message Signature ; Proof of Work ; Nonce ; Asymmetric Encryption ; **Key ; Mining ; Consensus** ; **Private Key ; Decryption** ; **Message Hash**

Vocabulary Term	Definition
Asymmetric Encryption	When different keys are used to encrypt and decrypt a message
Consensus	A mechanism requiring enough people to agree which transactions are valid
Decryption -	Unscrambling a message to make it readable
Encryption	Scrambling or changing a message to hide the original text
Key	A secret password for unlocking a message
Ledger	A record of all transactions in a group
Message Hash	a unique representation of an original message that has been transformed so it is unrecognizable.
Message Signature	A unique encrypted message used to verify the sender of a message.
Mining	When you continually generate new numbers to try and get a hash to start with a unique set of 0's. This is a puzzle that can only be solved by guessing
Nonce	A number used to try and change the hash of a block of messages so that it starts with a unique set of 0's. It's an abbreviation for "Number used once"
Private Key	A key that is kept private so only a specific person can decrypt a message
Proof of Work	Verifying information with a lot of computing effort
Public Key	A key that is shared with everybody so anyone can encrypt a message
Symmetric Encryption	When the same key is used to encrypt and decrypt a message

3- Verifying Signatures

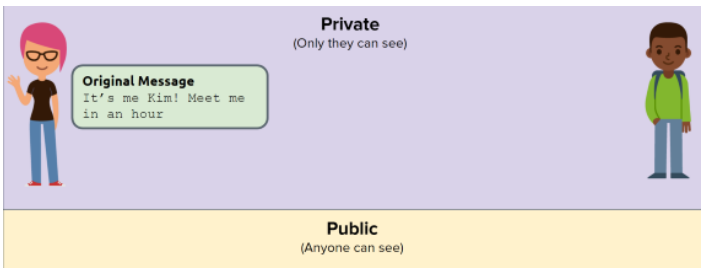
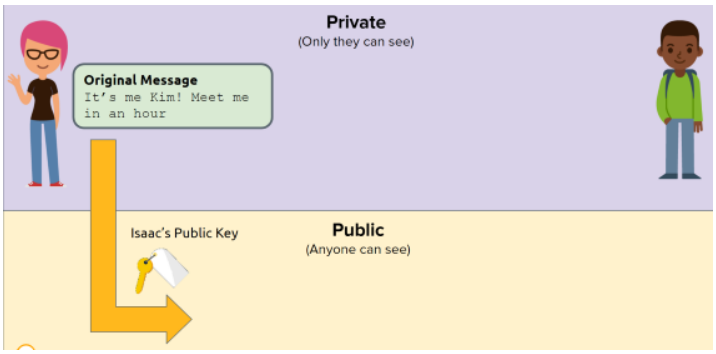
The blockchain helps facilitate thousands of transactions each day between people who've never met or trust each other.

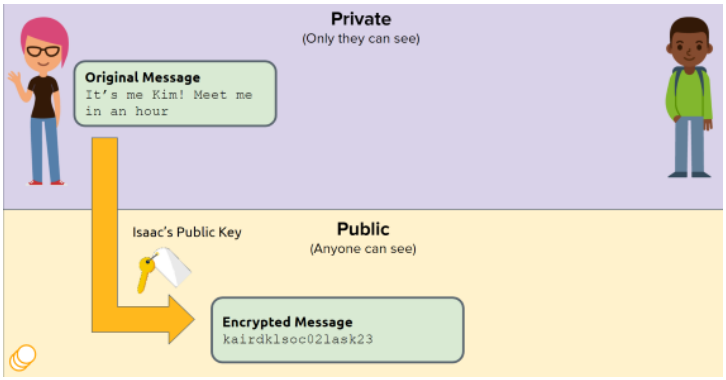
Cryptographic signatures help verify that the person is actually who they say they are.

https://www.youtube.com/watch?v=Ssw63fBF20g

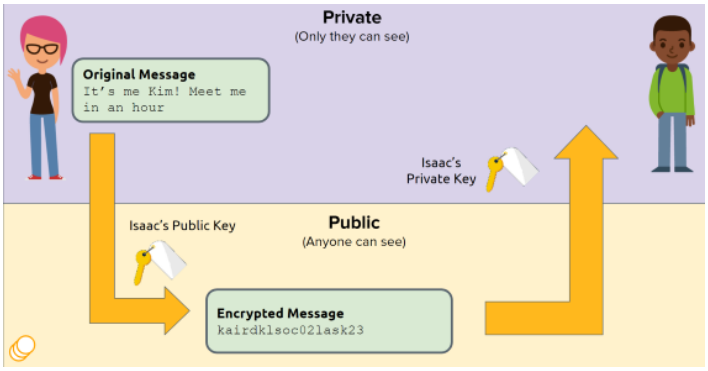
Public Key Encryption

Here is an example showing how a message can be encrypted and decrypted using public and private keys.

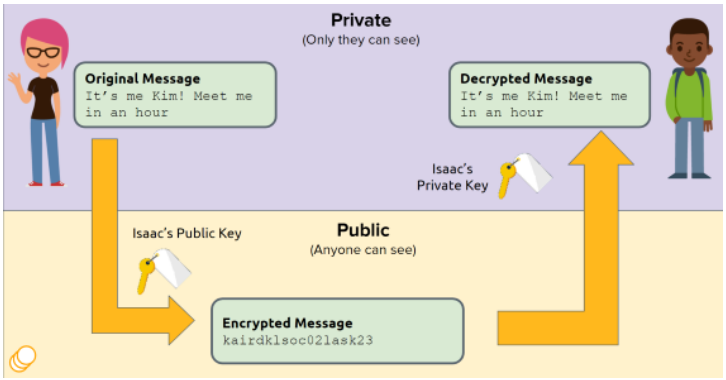
	In this example, Kim wants to send a secret message to Isaac.
	She starts by taking Isaac's **public** key and using it to encrypt her message.



The result is a string of letters and numbers that were specially created by Isaac's **public** key. Kim sends this to Isaac.

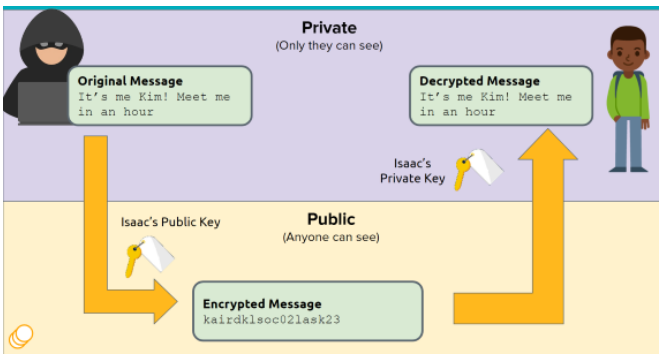


In order to retrieve the original message, Isaac takes the encrypted message and uses his secret **private** key to decrypt it.



When Isaac decrypts with the **private** key, he reverses the encryption created by the **public** key and ends up with the original message. This is only possible because the two keys are created especially for this purpose.

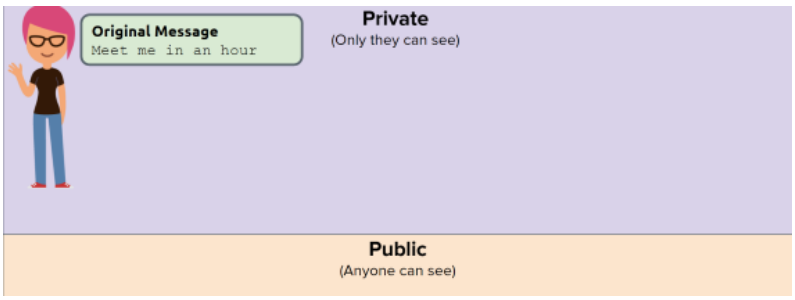
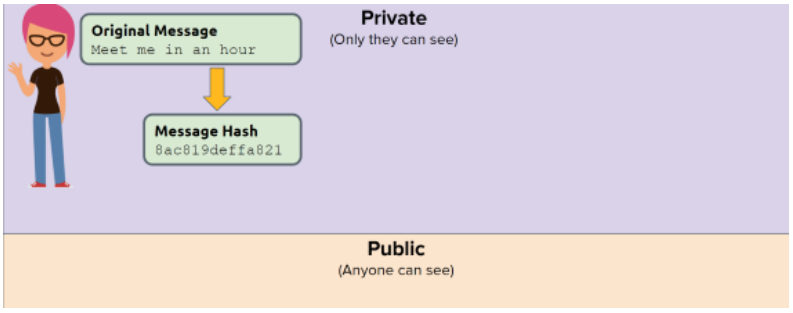
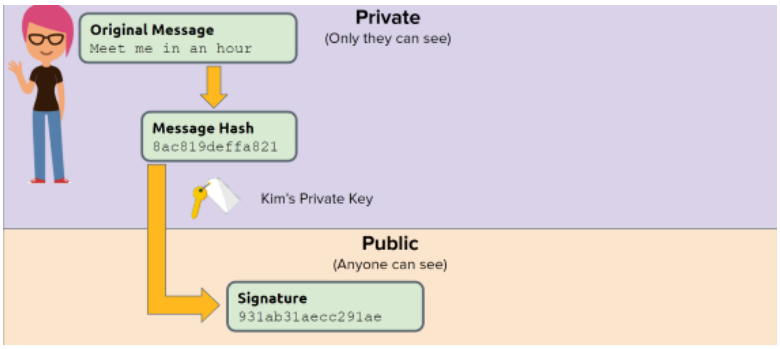
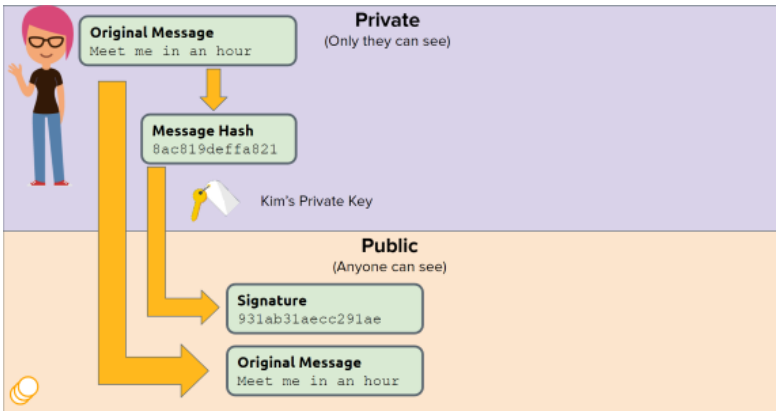
Encryption is great for making sure that no one else can read the messages you are sending. But there still could be a problem: how do we know for sure that the person sending the message is who they say they are?

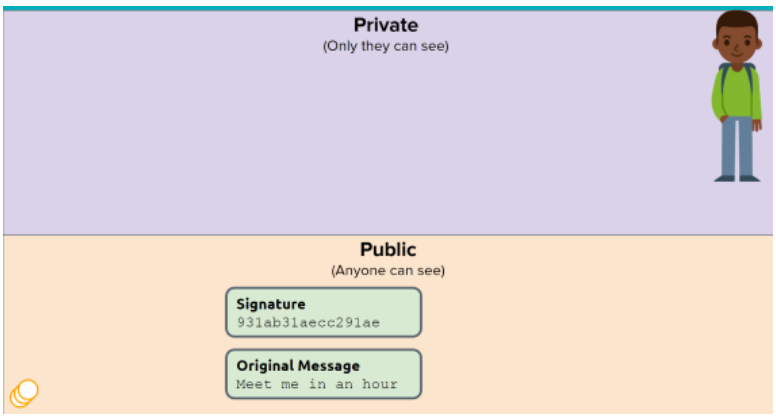
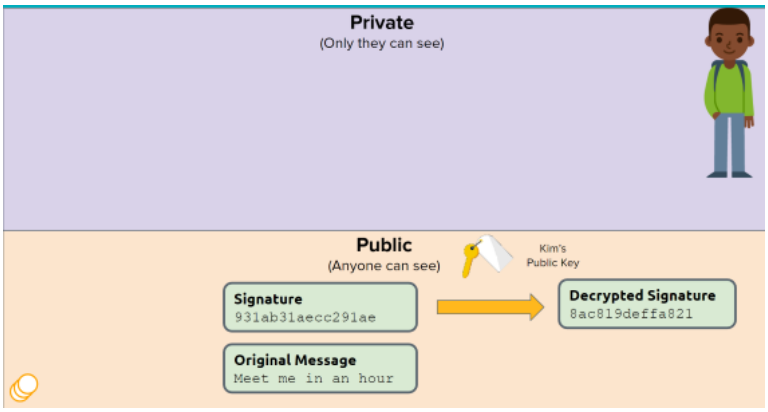
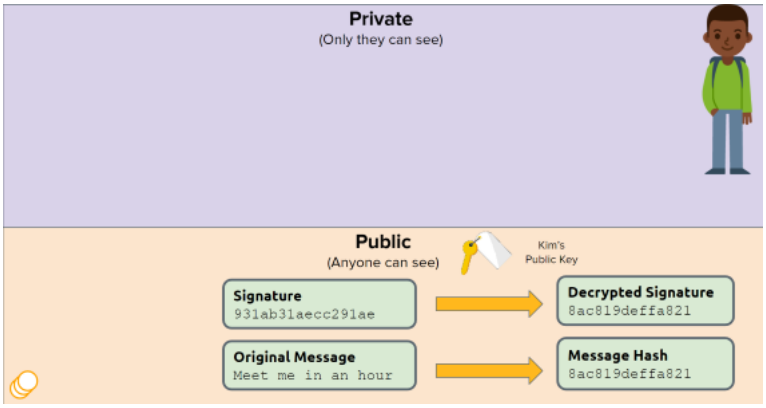
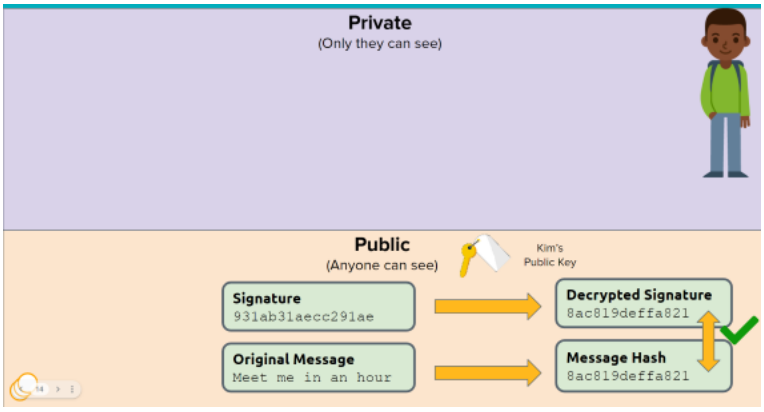


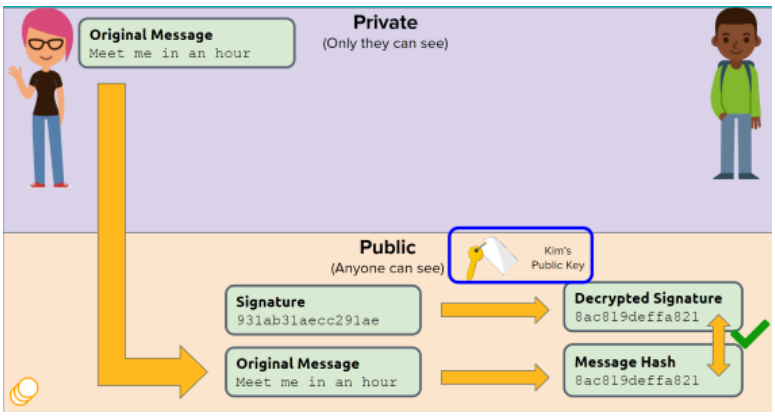
How can we make sure the person we're talking to is actually who they say they are?

Trusting the person that we're talking to requires two things: a unique way to represent the message to make sure it wasn't forged, and a way to verify the person who sent the message. The first part - representing messages in a unique way - can be solved with something called a message hash.

This is how public key cryptography can be used to verify the sender using a message signature.

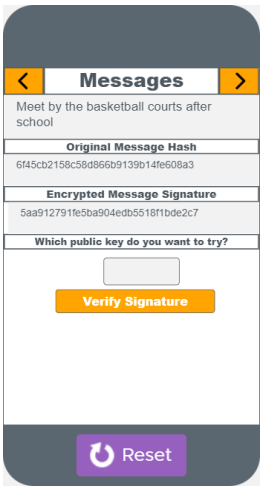
	Let's say Kim wants to send Isaac a message and prove that it came from her.
	First, she converts the message into its message hash using a special process. The thing that makes a hash special is that it is unique to the original message.
	Then Kim takes her **private** key and uses that to encrypt the message hash, creating another string of letters and numbers. This is the **signature**, which was created with the private key that only Kim has.
	Now Kim is ready to send her message, which is now two parts: the original message and the signature. In this example, Kim isn't using any additional encryption but she could if she wanted to.

	Now we switch to Isaac's perspective. Right now, Isaac can see both the signature and the message but he doesn't know for sure that Kim sent it. Here are three steps he needs to take to verify if she sent the message.
	Step 1 - he takes Kim's **public** key and uses it to decrypt the **signature**. This reverses the encryption created by Kim's **private** key.
	Step 2 - Isaac takes the message he received and determines its hash using the same process Kim did.
	Step 3 - Finally, Isaac compares the results from his two calculations to see if they match. In this case, they do!



Since Isaac used Kim's public key to verify the signature, that means she was the only person who could've sent the message.

The widget : verifying signatures



https://studio.code.org/s/blockchain-2023/lessons/2/levels/1?no_redirect=1

Several people have sent messages to the group, but we're not sure who sent which message! Luckily, each message was **signed** with someone's private key. To figure out who sent the message, we can use the public keys to verify the signature and determine who sent the message. This involves de-crypting the message signature to see if it matches the original message - if it does, then we know who was the authentic author of the message!

Do This

- Use the widget on the left to view the messages. You can use the orange arrows at the top to scroll between messages. There are four messages in total 

- Choose a public key from the table below to try and decrypt the message signature
  - If the decrypted signature matches the original message: you've found the sender!
  - If not, then try another person's public key
- Repeat this process until all messages have been verified!

Use the widget on Code Studio to fill in the table below with the correct sender of each message

Message	Sender
Meet by the basketball courts after school	Kim
Let's hang out this weekend	Isaac
I brought some empanadas to share at lunch	Hawa
Someone told me Zoey is the coolest person in school	Zoey

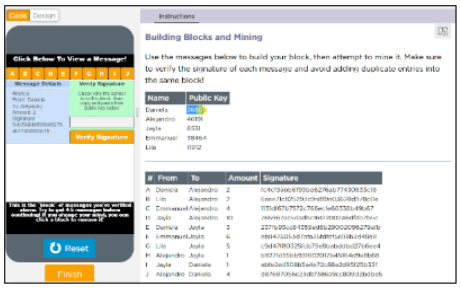
4- Blockchain and Bitcoin

Using encryption is important when trust can be an issue - for example, sending a secret message and not wanting someone to eavesdrop on it. Another situation where trust can be an issue is with money and finances - how can two people exchange goods without needing to trust each other? This is a problem that the Blockchain and Bitcoin have been able to solve

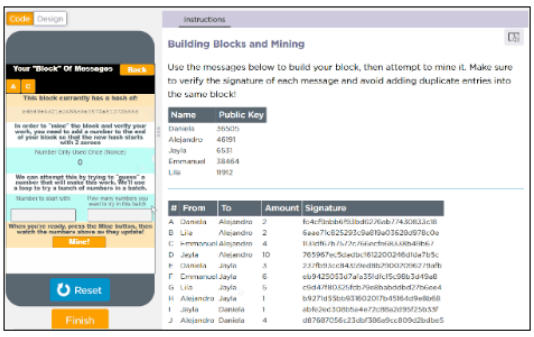
Blockchain technology helps solve the problem of exchanging money without relying on a shared sense of trust - instead, people trust the underlying technology: encryption. However, there is still the need to verify the transactions and add those transactions to the blockchain. This is what the blockchain miners do. Let's see if we can simulate what it would be like to be a miner on the blockchain.

The widget : mining blocks

https://studio.code.org/s/blockchain-2023/lessons/2/levels/2?no_redirect=1

Widget	Modeling Notes
	• Click on each of the buttons to load a message. Each message represents a bitcoin transaction. • Identify the sender of the message and find their public key in the table. Copy and paste their public key into the widget to verify the signature • If the signature matches: you can add that message to your "block" by clicking the "Add To Block" button • If the signatures don't match: you can ignore that message since it's a fraudulent message. • As a class, work together to add 3-5 blocks, then press the Next Screen button.

Trying to find the correct nonce can only be done with guessing and trying lots of combinations, which is what makes this a proof of work. Let's look at how we can do this in the widget.

Widget	Modeling Notes
	• When the level loads, it will automatically reveal the steps to begin mining and completing a proof of work • Choose a starting number to test for your nonce. You can ask students to suggest this number • Choose how many numbers to try. You can also ask students to suggest this number. • Press the Mine button and see if a match is found! • If no match is found, choose a new starting number or a new range to test and try again!

Blockchain Scenario #1

Use the widget on Code Studio to record your work building a block on this particular blockchain.

Which messages were valid?	Which messages were invalid?
A, C, D, E, G, H, J	B, F, I
Which messages did you include in your block?	**What was the nonce you found**
[One possible answer] A, C, D, E, G, H, J	96

Since the transactions are public, we can see how these people are trading money. Does it look like this scenario has a pattern? If so, what story can you tell about that scenario amongst this group of friends?

A group of friends paying each other back for meals when they can't split the check and 1 person has to pay for everyone (similar to Venmo or Zelle transactions)

Blockchain Scenario #2

Use the widget on Code Studio to record your work building a block on this particular blockchain.

Which messages were valid?	Which messages were invalid?
A, B, C, D, E, F	G, H, I, J
Which messages did you include in your block?	**What was the nonce you found**
[One possible answer] A, B, C, D, E, F	302

Since the transactions are public, we can see how these people are trading money. Does it look like this scenario has a pattern? If so, what story can you tell about that scenario amongst this group of friends?

One person trying to create fraudulent transactions and steal money from others

Blockchain Scenario #3

Use the widget on Code Studio to record your work building a block on this particular blockchain.

Which messages were valid?	Which messages were invalid?
A, B, C, D, E, F, G, H, I	J
Which messages did you include in your block?	**What was the nonce you found**
[One possible answer] A, B, C, D, E, F, G, H, I	258

Since the transactions are public, we can see how these people are trading money. Does it look like this scenario has a pattern? If so, what story can you tell about that scenario amongst this group of friends?

Some kind of gambling ring, where everyone sends their “bets” to a central person who then redistributes it to the winner

Blockchain Scenario #4

Use the widget on Code Studio to record your work building a block on this particular blockchain.

Which messages were valid?	Which messages were invalid?
A, B, C, D, E, F, H, I	G, J
Which messages did you include in your block?	**What was the nonce you found**
[One possible answer] A, B, C, D, E, F, H, I	156

Since the transactions are public, we can see how these people are trading money. Does it look like this scenario has a pattern? If so, what story can you tell about that scenario amongst this group of friends?

One person gives “loans” to their friends, who pay it back later with a little extra as interest.