Ethereum introduction

**Ethereum and smart contracts introduction:**

* Ethereum is a blockchain 2.0 technology, this means thatg we have dynamic and programmable blockchains.
* Internett started with static websites, only text images and links (web 1.0). web2.0 happened because we added javascript which lead to dynamic and programmable websites.
* Blockchain before ethereum is like internet without javascript.
* Smart contracts: small pieces of code that run on the ethereum blockchain, we can make these contracts thtough the programming language solidity. This smart contract is unstoppable (cant be sutdown), can see what is hapening with it in terms of transactions. Tey can own and then decide what to do with money. Program it to for example make my children inherit all my money, i send the money to the smart contract, and it cant be taken out before the programmed action happens, for example after 10 years send this money to my children. No need for third parties. **Programmable money.**
* It gives transparency because everyone can see how sends and recieves money, free from governments, and therefore free for corruption.
* Nobody can go back and change how a smart contract behaves, it is integrated into the blockchain. If you try to change it you willm have the same problem as with trying to remove a bitcoin transaction.

**EVM Solidity and smart contracts:**

* Ethereum is a network of computers.
* Ethereum is using the same consensus algorithm as bitcoin, they have miners that use PoW aswell.
* Ethereum adds another layer to the bitcoin system which is EVM, makes it possible to execute code on the network in such a way that we can reach consensus.
* Behind the scenes: a simple program f.eks. skype are just instructions that normaly your cpu would execute. CPU is interacting with the RAM GPU screen and so on. Inn ethereum when you are executing smart contracts there is a network of computers and not a single CPU exectuing it. Of course every computer in this network has a CPU, GPU and so on. Ethereum virtual machine (EVM) executes these codes on all the computers. It is a piece of code thatg all the computers on the network are running. Every computer has the EVM, and it runes on top of the existing hardware of the node. The EVM can also read and execute smart contracts. So now the CPU isnt executing the code directly, only indirectly through the EVM, will tell the CPU what to do. All the different nodes need to execute the instructions given by the EVM so they reach the same result and then we reach a consensus about what happened in the smart contract.
* The reason why the CPU cant directly execute the code is because of security. Anyone can deploy a smart contract onto the ethereum blockchain. Maybe the smart contract is programmed to read personal information on your computer, and if your CPU directly executes that instruction, then it can be dangerous. Therefore the EVM will insure that nothing mallisus happens
* If someone can break through the EVM then the entire ethereum network is compromised.
* How do the EVM execute instruction:?

Solidity code will be complied into byte code and then the EVM will read the byte code instructions. Potentially you can have other programming languages as long as you can compile the code into byte code, working on a programming code called viper.

* How ethereum started:?

Wanted to make this happen on the bitcoin blockchain so it could be programmable and dynamic. Bitcoin has scrpit as a langugae, but it is very primitive, Bitcoin is not turing complete, cant solve all the problems you want it to solve. SOlidity in ethereum is turing complete everything can be programmed as you want it.

* Loops is a feature that allows you to tell the software to do one thing several times. Without loops you need to write it all by hand.
* But loops can also create alot of spam on the blockchain which makes the blockchain alot larger with all the smart contracts being deployed

**Contents of the nodes in ethereum blockchain:**

* In bitcoin the nodes have a copy of the blockchain with all the transactions.
* In Ethereum te nodes have the blockchain with smart contracts, and all transactions, and the code for all ethereum smart contracts, then you also have the state of the samrt contracts. Transactions look a bit differently because they have no UTXOs,
* The smart cotnracts can be in different states. The first state if for example when the smart contract is empty, when i send money to the smart con tract that could be state 2, in state 3 the kid has finished school and checks the kids grades, and in the last state the smart contract either pays the kid or give me the money back. And then we og back to state 1.
* The state of the smart contracts could last for years.

**UTXO vs account model – privacy and fungibility**

* In bitcoin you have UTXOs according to input and outputs, and therefore you can track the transaction. In bitcoin your wallet calculates all UTXOs that you can spend and add them together and get the balance.
* On Ethereum you have a balance and it will store all transaction without UTXOs. And the transactions will upadte the wallets connected tp the transactions. For example take 1 ethereum away from your wallet and give 1 ethereum to another persons wallet.
* «Move founds from this account to thatg account»
* The transactions therefore take less space, because you dont have to worry about all the inputs and putputs and different UTXOs, and you dont have to send transactions back to yourelf.
* In ethereum an ethereum transaction you cant have several inputs and outputs, gives less privacy but way easier to track which wallet gets the money.
* Harder to trace the coins in terms of going several transactions back in time and all theh way back to the miner

Homework:

Ethereum introduction:

1. Web 1.0 was when we only had static websites, with texts, imgaes and links.
2. A smart contractt is a piece of code (solidity) on the ethereum blockchain. This piece of code can handle money according to the code (instructions) you have givent he smart contract. For example you have programmed the contract to give money to your children when they get good grades in school. Then you send money to the contract and the money will only be moved if they get good grades at school. They are unstoppable.
3. Transparency is acheived with ehtereum code because everyone can see the transactions happening through the smart contracts on the blockchain.

EVM:

1. All code written on the ethereum network needs to og through the EVM first. The EVM is «above» your computers hardware and will tell all the computers (nodes) CPUs what code they are supposed to execute.
2. The format of the instructions the EVM can read is byte code.
3. The realationship between solidity and byte code is that solidity code can be compiled into byte code
4. Bitcoin cant run the same typem of complex programs as ethereum because they dont have a turing complete language and the language they use is called script. If bitcoin used a turing complete language like solidity, people could create loops with the code which would create a lot of spam on the blockchain causing it to take up a lot more space.
5. A turing complete language is a coding language where you can solve all problems through code, it is not limited.

UTXO and account model:

1. The benefits of the account model not requiring as much space as the UTXO model is that you will have space for more transactions per block, which in turn would reduce the transaction fees. The requirements for the nodes storage space can be lower than it would have to be with the UTXO model, thus making it cheaper to mine=more miners=better hash power= better security.
2. The account model is not as great for privacy because you can only send money from one wallet to another (only one input and one output) this makes it a lot easier to trace the money, therefore knowing exactly where the money was sent from and to whom it was sent to. You cant for example send founds back to yourself aswell as to another person like you can in the UTXO model.