Blockchain Questions Bank

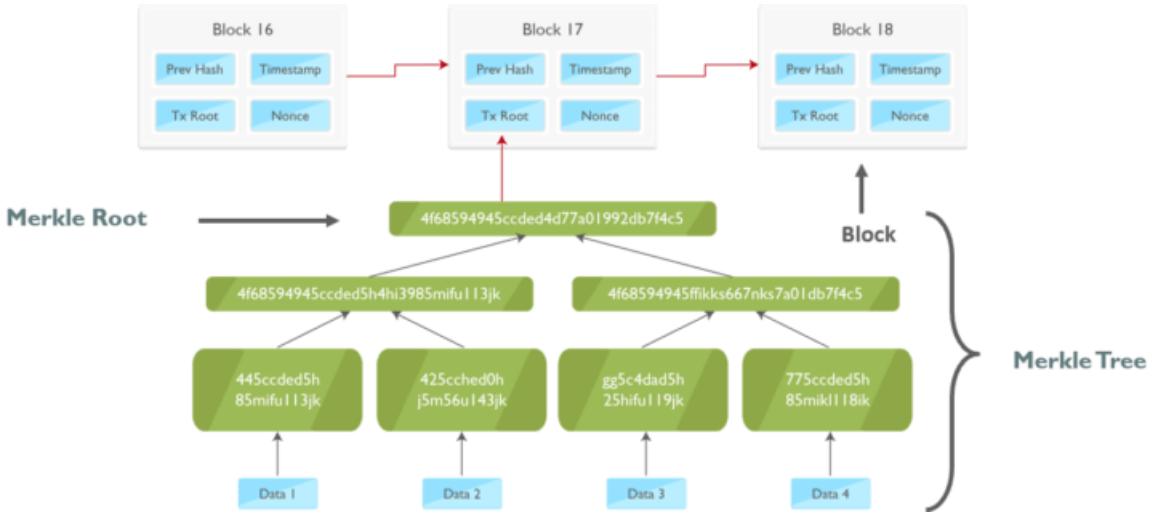
1. Differentiate between Blockchain and Hyperledger.

Blockchain is a decentralized technology of immutable records called blocks, which are secured using cryptography. Hyperledger is a platform or an organization that allows people to build private Blockchain. Using Blockchain you can build public and private Blockchain whereas with Hyperledger you can only build private Blockchains.

2. What is Merkel Tree?

Merkel Tree is a data structure that is used for verifying a block. It is in the form of a binary tree containing cryptographic hashes of each block. A Merkle tree is structured similarly to a binary tree where each leaf node is a hash of a block of transactional data and each non-leaf node is a hash of its leaf node. The Merkel root or hash root is the final hash root of all the transaction hashes. It encompasses all the transactions that are underlying all the non-leaf nodes.

3. What do you mean by blocks in Blockchain technology?



Blockchain is a distributed database of immutable records called blocks, which are secured using cryptography. There are a previous hash, transaction details, nonce, and target hash value. A block is like a record of the transaction. Each time a block is verified, it gets recorded in chronological order in the main Blockchain. Once the data is recorded, it cannot be modified.

4. What is cryptography? What is its role in Blockchain?

Blockchain uses cryptography to secure users' identities and ensure transactions are done safely with a hash function. Cryptography uses public and private keys in order to encrypt and decrypt data. In the

Blockchain network, a public key can be shared with all the Bitcoin users but a private key (just like a password) is kept secret with the users.

5. What are the different types of Blockchain?

There are different types of Blockchain - Public, Private, Consortium and Hybrid Blockchain.

- Public Blockchain ledgers are visible to all the users on the internet and any user can verify and add a block of transactions to the Blockchain. Examples, Bitcoin, and Ethereum.
- Private Blockchain ledgers are visible to users on the internet but only specific users in the organization can verify and add transactions. It's a permissioned blockchain, although the information is available publicly, the controllers of the information are within the organization and are predetermined. Example, Blockstack.
- In Consortium Blockchain, the consensus process is controlled by only specific nodes. However, ledgers are visible to all participants in the consortium Blockchain. Example, Ripple.
- A hybrid blockchain is a unique integration of two different types of blockchains: public and private (permissioned) blockchains. It seeks to combine the advantages of both while addressing some of their individual limitations. In a hybrid blockchain architecture, certain aspects of the blockchain network are public and decentralized, while others remain private and controlled by designated entities.

6. What is a Genesis Block?

The genesis block is the first block in the Blockchain which is also known as block 0, it is the only block that doesn't refer to its previous block.

7. Where do nodes run a smart contract code?

Nodes run smart contracts code on Ethereum Virtual Machine (EVM). It is a virtual machine designed to operate as a runtime environment for Ethereum-based smart contracts.

8. What is a Dapp and how is it different from a normal application?

Dapp:

- A Dapp is a decentralized application which is deployed using smart contract
- A Dapp has its back-end code (smart contract) which runs on a decentralized peer-to-peer network

Normal application:

- Normal application has a back-end code which runs on a centralized server
- It's a computer software application that is hosted on a central server

9. Are there any network-specific conditions for using Blockchain technology in an organization?

There is no specific network condition, but the network must be a peer-to-peer network under the concerned protocols.

10. What is the nonce and how is it used in mining?

In Blockchain, mining is a process to validate transactions by solving a difficult mathematical puzzle called proof of work. Now, proof of work is the process to determine a number that used only once (nonce) along with a cryptographic hash algorithm to produce a hash value lower than a predefined target. The nonce is a random value that is used to vary the value of hash so that the final hash value meets the hash conditions.

11. How are the blocks and transactions encrypted in a bitcoin implementation?

Every block in a bitcoin implementation is a public block, so the blocks are not encrypted in any way. Block content is processed using a special hash function, SHA-256 to prevent modification and guarantee data integrity. This block hash value is included in the blockchain.

12. What is the fork? What are some of the types of forking?

In simple terms, updating a cryptocurrency protocol or code is called forking. Fork implies that a Blockchain splits into two branches. It can happen when the participants of the network cannot come to an agreement with regards to the consensus algorithm and new rules to validate transactions.

There are three types of forking:

- Hard forks
- Soft forks

13. Differentiate between Proof of Work vs Proof of Stake.

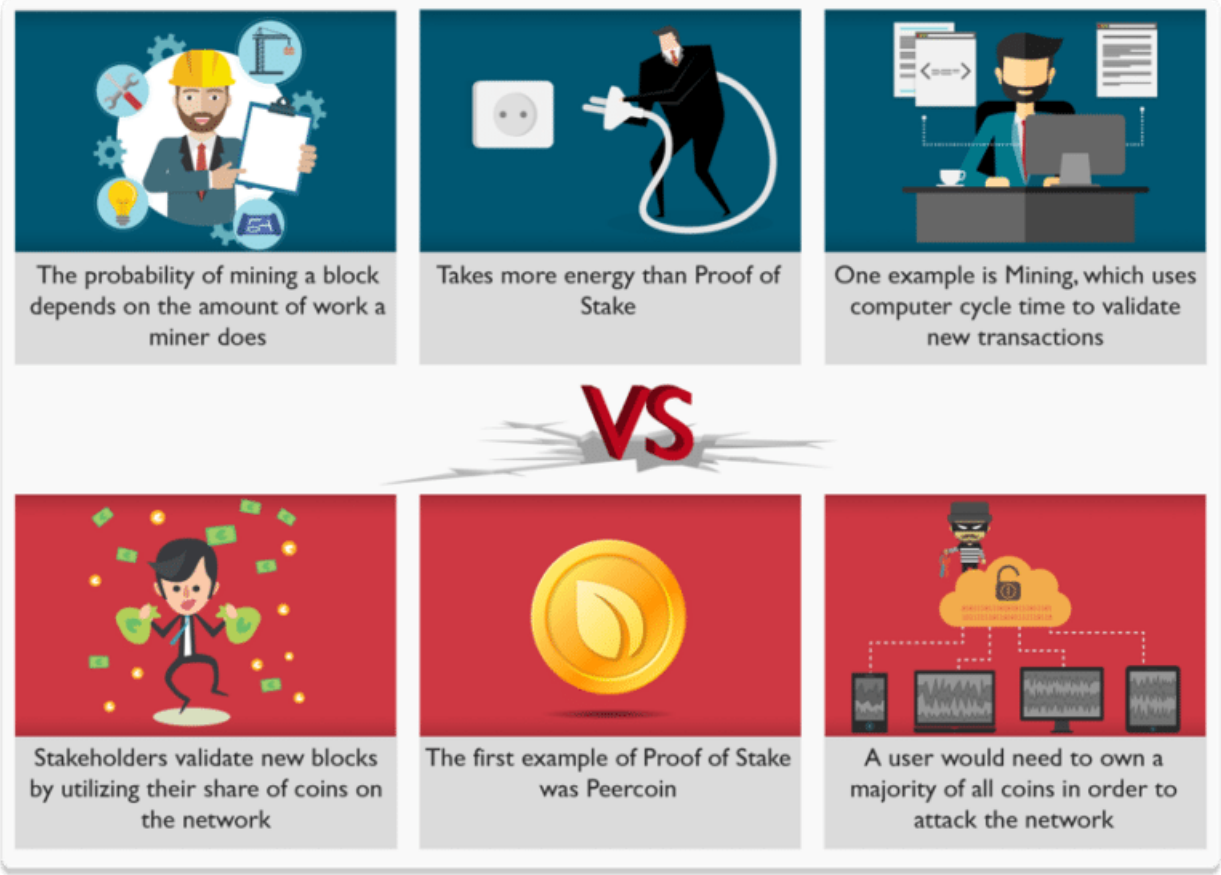
Proof of Work (PoW):

In Blockchain, PoW is the process of solving a complex mathematical puzzle called mining. Here, the probability of mining a block is based upon the amount of computational work done by a miner. Miners spend a lot of computing power (with hardware) for solving the cryptographic puzzle.

Proof of Stake (PoS):

PoS is an alternative to PoW in which the Blockchain aims to achieve distributed consensus. The probability of validating a block relies upon the number of tokens you own. The more tokens you have, the more chances you get to validate a block. It was created as a solution to minimize the use of expensive resources spent in mining.

14. What is a 51% attack?



In Blockchain, a 51% attack refers to a vulnerability where an individual or group of people controls the majority of the mining power (hash rate). This allows attackers to prevent new transactions from being confirmed. Further, they can double-spend the coins. In a 51% attack, smaller cryptocurrencies are being attacked.

15. How to check if a block is a valid block?

When a new block is announced on the network, every node that receives it does a list of checks. The two most important checks are:

- **Proof of work:** To check if a block provides enough work to be included in the chain.
- **Validity of all the transactions:** Each transaction must be a valid transaction.

16. What is the principle on which blockchain technology is based on?

It enables the information to be distributed among the users without being copied.

17. Why is Blockchain a trusted approach?

- Its compatibility with other business applications due to its open-source nature.
- Its security. As it was meant for online transactions, the developers have paid special attention to keeping up the pace when it comes to its security.
- It really doesn't matter what type of business one owns, Blockchain can easily be considered.

18. Name the two types of records that are present in the blockchain database?

These records are **block records** and **transactional records**. Both these records can easily be accessed, and the best thing is, it is possible to integrate them with each other without following the complex algorithms.

19. Blockchain is a distributed database. How does it differ from traditional databases?

Properties	Blockchain	Traditional Database
Operations	Only Insert Operations	Can perform C.R.U.D. operations
Replication	Full Replication of block on every peer	Master Slave Multi-Master
Consensus	Majority of peers agree on the outcome of transactions	Distributed Transactions (2 phase commit)
Invariants	Anybody can validate transactions across the network	Integrity Constraints
Centralization	Decentralized	Centralized

20. What do you mean by blocks in blockchain technology?

Blockchain consists of a list of records. Such records are stored in blocks. These blocks are in turn linked with other blocks and hence constitute a chain called Blockchain.

21. How does a block is recognized in the Blockchain approach?

Every block in this online ledger basically consists of a hash pointer which acts as a link to the block which is prior to it, transaction data and in fact a stamp of time.

22. Is it possible to modify the data once it is written in a block?

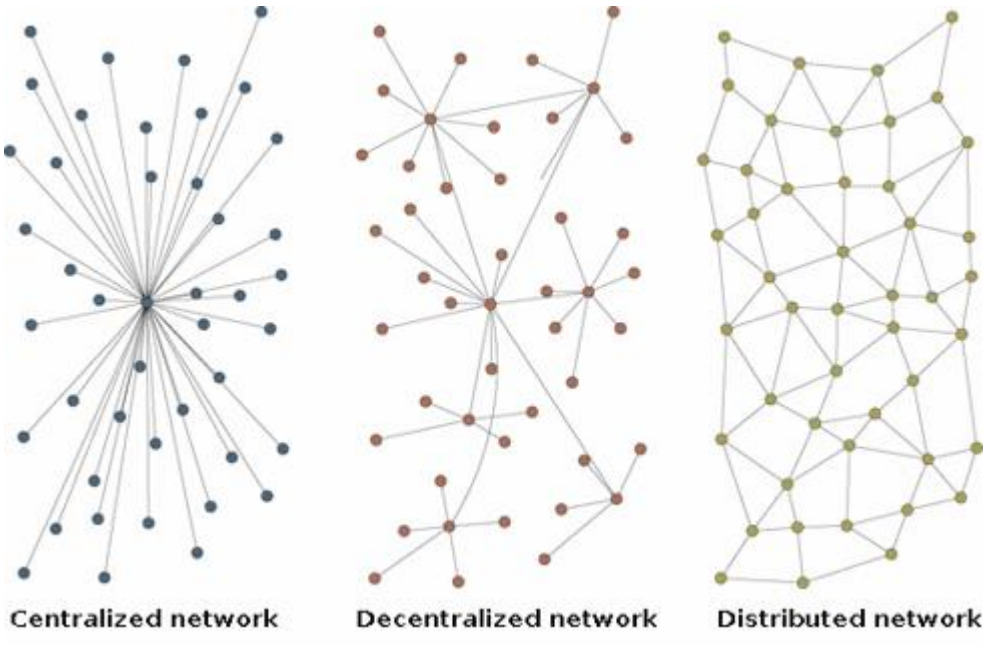
No, it's not possible to do so. In case any modification is required, the organization simply has to modify the information on all other blocks too.

23. What are Block Identifiers?

In Blockchain, blocks can be identified by the *block header hash*.

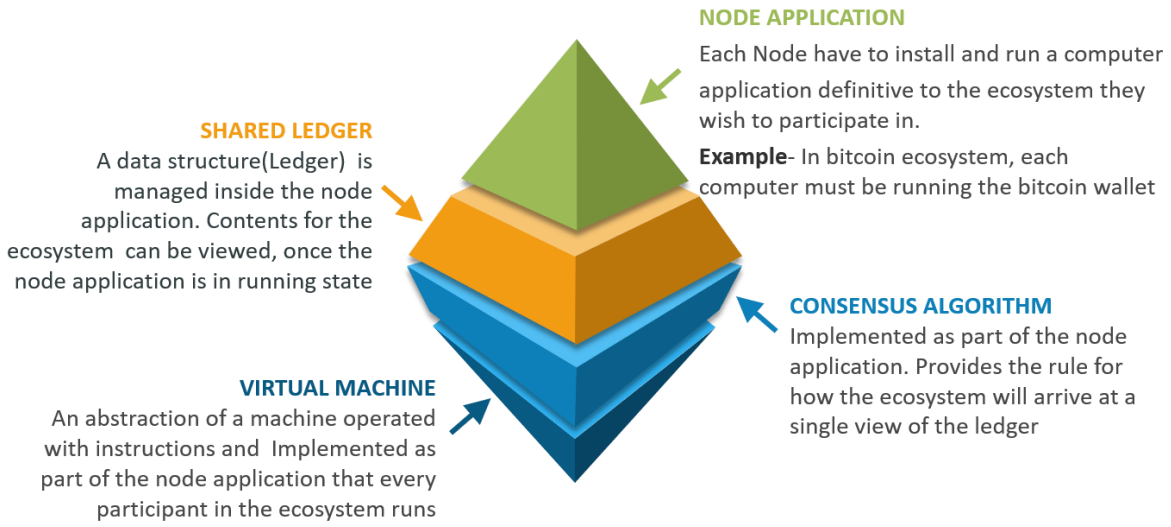
24. Name the common type of ledgers that can be considered by users in Blockchain?

These are:



25. Can You explain the components of Blockchain Ecosystem?

Following are the components of blockchain Ecosystem:



26. What is Double Spending? Is it possible to double spend in a Blockchain system?

It's a condition when one digital token is spent multiple times because the token generally consists of a digital file that can easily be cloned. It simply leads to inflation and organizations must bear a huge loss. One of the primary aims of Blockchain technology is to eliminate this approach up to the possible extent.

Blockchain prevents double spending by confirming a transaction by multiple parties before the actual transaction is written to the ledger. It's no exaggeration to say that the entirety of bitcoin's system of

Blockchain, mining, proof of work, difficulty etc, exist to produce this history of transactions that is computationally impractical to modify.

27. Can you explain what are off-chain transactions?

An off-chain transaction is the movement of value outside of the blockchain. While an on-chain transaction – usually referred to as simply ‘a transaction’ – modifies the blockchain and depends on the blockchain to determine its validity an off-chain transaction relies on other methods to record and validate the transaction.

28. Name the steps that are involved in the Blockchain Block Generation?

1. Transaction preparation
2. Transaction verification
3. Block generation
4. Block validation
5. Block chained

29. What is a hard fork?

A hard fork is a change to the underlying consensus rules or protocol that results in the creation of a new, independent blockchain branch. The term "hard fork" is used to refer to the permanent divergence in the blockchain network due to incompatibilities between the old rules and the new rules.

In the context of Ethereum, a hard fork may occur when there is a change to the Ethereum protocol that requires all participants in the network (miners, nodes, and clients) to update their software. Participants who do not upgrade will remain on the old chain, while those who do will create a new chain with the updated rules.

30. What is a soft fork?

A soft fork occurs when a blockchain undergoes a backward-compatible update to the protocol or rules. Soft forks tighten the existing rules or introduce new ones but maintain compatibility with the previous set of rules. To be more precise, the term "soft fork" is not specific to Solidity but applies to any blockchain system, including Ethereum, which Solidity is built upon.

For instance, a change in the Ethereum network's consensus algorithm, applied through a soft fork, would still allow nodes that have not upgraded to the new version to validate and accept blocks generated by nodes using the updated rules. However, the nodes that have not upgraded might still produce blocks that may be considered invalid by up-to-date nodes.

31. What is Proof of Work?

It is a computationally intensive process that requires miners to solve complex mathematical problems to add new blocks to the blockchain.

32. what is pow difficulty?

Defined by the number of leading zeros Hash output requires to solve proof of work

33. what is the different types of nodes?

Full Nodes – Store the entire blockchain history and have the ability to validate all transactions independently.

Pruning Nodes – tore only a portion of the blockchain history (e.g., recent data), keeping older blocks compressed or removed after validation.

Lightweight Nodes – Only store blockchain headers (not full blocks) and rely on full nodes for transaction verification.

34. What is Nodes?

Nodes are computers that participate in the blockchain network. They store blockchain data and verify new transactions.

Or A software client that participates in the network.

35. What are Transactions?

are the basic unit of interaction in a blockchain. They can be used to transfer value, create contracts, or record other events.

36. what is digital signature?

A short string of data a user produces for a document using a private key such that anyone with the corresponding public key, the signature, and the document can verify that (1) the document was "signed" by the owner of that particular private key, and (2) the document was not changed after it was signed.

37. what is Tokenization?

the process of representing real-world or digital assets as tokens on a blockchain network. It involves converting the ownership or rights of an asset into a digital representation, which can be traded, transferred, and recorded securely on the blockchain

38. what is Consensus?

The mechanism used to agree on the state of Blockchain.

Multiple Choice Blockchain Questions

Q 1. Each block of a Blockchain consists of which of the following?

1. A hash pointer to the previous block
2. Timestamp

3. List of transactions
4. All of the above

Q 2. Which of the following is first distributed blockchain implementation?

1. Bitcoin
2. Ethereum

Q 3. Bitcoin is based on _____ blockchain?

1. Private
2. Public
3. Public Permissioned
4. Permissioned

Q 4. Blockchain can be stored as which of the following?

1. A flat file
2. A Database
3. Both of the above
4. None of the above

Q 5. In blockchain, blocks are linked _____?

1. Backward to the previous block
2. Forward to next block
3. Not linked with each other

Q.6 In block chain, _____ tree stores all the transactions in a block by producing a digital fingerprint of the entire set of transactions.

1. Merkle
2. Binary
3. AVL
4. Red black
5. None of the above/More than one of the above

Q 7. Hash identifying each block in the Blockchain is generated using which of the following cryptographic algorithm?

1. SHA128
2. SHA256

Q 8. A block in the blockchain can never have more than one parent block?

1. True
2. False

Q 9. Blockchain forks can result in which of the following?

1. Multiple parent blocks
2. Multiple children blocks

Q 10. Which of the following is asymmetric encryption Algorithm?

1. Blowfish
2. Twofish
3. RSA
4. Tripple DEA

Q.11 What is a blockchain?

1. A blockchain is a centralized digital ledger consisting of records called blocks.
2. A blockchain is a decentralized, distributed, digital ledger consisting of records called blocks.
3. A blockchain is a digital database consisting of records called class.
4. None of the above

Q.12 The term used for a blockchain splits is _____.

1) A merger
2) A fork
3) A division
4) None of the above

Q.13 What are the pillars of blockchain technology?

1. Transparency
2. Immutability
3. Decentralization
4. All of the Above

Solidity Questions

1. What is a smart contract?

Smart contracts are self-executing digital contract with the terms of the agreement directly written into code which run on a blockchain network.

2. What is a gas limit in Solidity?

A gas limit in Solidity refers to the maximum amount of gas a user is willing to spend on a transaction.

3. What is a variable in Solidity and its types?

A variable in Solidity is a storage location that can contain values. These values can be changed during runtime. Variables are broadly classified as state variables, local variables and global variables.

4. What is an event in Solidity?

An event is an inheritable member of the contract, which stores the arguments passed in the transaction logs when emitted.

5. What is the constant function in Solidity?

A constant function does not change the state of the contract.

6. How you could Store a public key in Solidity?

Using the address data type.

7. List 3 different types of Ether units available in Solidity?

wei, kwei, gwei.

8. What is the different between storage and memory variables?

Parameters	Memory	Storage
Definition	Memory in Solidity is a temporary place to store data.	Storage holds data between function calls.
Memory Type	The Solidity Smart Contract can use any amount of memory during the execution but once the execution stops, the Memory is completely wiped off for the next execution.	Storage is persistent, each execution of the Smart contract has access to the data previously stored in the storage area.
Usage	It is better to use Memory for intermediate calculations.	It is better to use Storage to store the final result.
What Does it Store?	Function arguments are stored in Memory.	State variables and Local Variables of structs, and arrays are always stored in Storage by default.

9. What is the global variable used for the transaction sender?

Msg.sender

10. What is special in smart contracts compared to other programs?

- **Immutable:** Once the smart contracts are deployed on the blockchain, their code cannot be altered. This makes the smart contract invulnerable to unauthorized access.
- **Decentralized execution:** The execution of smart contracts does not depend on a single authority but instead on multiple nodes that are spread around the world.
- **Self-executing:** Smart contracts are designed to execute predefined actions automatically when certain conditions specified in the contract are met. This reduces the need for intermediaries thus reducing the potential for human error.
- **Tokenization:** Smart contracts can be used to create and manage digital assets through the use of tokens.

11. List the difference between uint8 and uint16.

- uint8 stores a number of up to 2^{8-1}. It has 8 bits.
- uint16 stores number up to 2^{16-1}. It has 16 bits.

12. What are private and public variables in Solidity?

- Private variables can be accessed only within the contract that declares them. These are not accessible by an external entity nor by the contracts derived from the contract in which they are declared.
- Public variables can be accessed by any contract, function, or external entity. When these variables are declared in Solidity, the compiler automatically generates a getter function that allows any external entity or contract to read its value.

13. How much is 1 gwei of Ether?

1 Gwei equals 0.000000001 or 10^{-9} ETH.

14. How much is 1 wei of Ether?

1 wei is equivalent to 10^{-18} ETH.

15. List the differences between view and pure functions.

Parameters	View Functions	Pure Functions
Definition	These are read-only functions, which ensures that state variables cannot be modified after calling them.	These do not read or modify the state variables, which return the values only using the parameters passed to the function or local variables present in it.

Parameters	View Functions	Pure Functions
Data Access	read	None
Transaction Type	Call	Call
Ideal For	Getter functions to view data on the blockchain.	Defined to the scope of the function and not view data on the blockchain.

16. What is the ABI of the contract?

The Application Binary Interface (ABI) allows anyone writing a smart contract to be able to communicate between a web application written in high-level programming language and the bytecode that is understandable by EVM.

- ABI defines how the contents of the library are stored inside the file and the program uses the ABI to search through the file and find what it needs.
- These act as function selectors defining the specific methods that can be called to a smart contract for execution.

17. List the difference between public and private visibility modifiers in Solidity?

Parameters	Public Visibility Modifier	Private Visibility Modifier
Accessibility	When a state variable, function, or contract is declared as public then it is accessible from any contract including the contract itself, derived contracts, and other external contracts.	When a state variable, function, or contract is declared as private then it is accessible only within the contract in which it is defined.
Accessibility by External Transactions	They can also be called directly by external transactions.	It cannot be accessed from external transactions.
Getter Function	Solidity automatically generates a getter function to allow external access.	No getter functions are generated for private state variables.

18. What is the largest value a uint256 can store?

uint256 is an unsigned integer that can hold a maximum value of 2^{256-1}. It requires 32 bytes of storage space.

19. What is a fallback function in Solidity?

A fallback function in Solidity is a special, unnamed function that gets executed whenever a contract receives Ether without any data or when a function call is made without specifying any function. It serves as the default function for a contract when no other function matches the given input. It must be marked external and payable if the contract is intended to receive Ether directly.

20. What is difference between revert and require?

revert consumes all the gas and reverts all state changes and operations, making it suitable for critical errors.

require refunds any unused gas and is often used for input validation and noncritical conditions.

21. Declare and Initialize an array of integers in solidity

```
pragma solidity ^0.8.0;

contract IntegerArray {
    // Declaring and initializing an array of unsigned integers
    uint[] public numbers = [1, 2, 3, 4, 5];
}
```

22. Declare a map which it's key is address mapped to boolean and make getters and setters function for it.

```
pragma solidity ^0.8.0;

contract AddressBooleanMapping {
    // Declare a mapping that maps addresses to booleans
    mapping(address => bool) public addressToBool;

    // Set a boolean value for a given address
    function setBoolForAddress(address _address, bool _boolValue) public {
        addressToBool[_address] = _boolValue;
    }

    // Get the boolean value of a given address
    function getBoolForAddress(address _address) public view returns (bool) {
        return addressToBool[_address];
    }
}
```

23. Write the code to add data to an array that has been declared as a state variable?

```
pragma solidity ^0.8.0;

contract ArrayExample {
    // Declare an array of unsigned integers as a state variable
    uint[] public myArray;

    // Function to add data to the array
    function addToMyArray(uint value) public {
        // Add value to the end of the array
        myArray.push(value);
    }
}
```

24. How can you add data to a mapping which is declared as a state variable?

```
pragma solidity ^0.8.0;

contract SimpleMapping {
    // Declare a mapping as a state variable
    mapping(uint => string) public identification;

    // Function to add data to the mapping
    function setIdentifier(uint id, string memory name) public {
        identification[id] = name;
    }
}
```

25. what is struct keyword used for?

It's used for user-defined data types.

26. What is the difference between constants and immutable variables?

The main difference is that constants are defined at compile-time, while immutable variables are defined at deployment time. It is possible to assign blockchain data to immutable variables but not to constants.

27. What does the gas usage in a transaction depend on and how is the transaction fee calculated?

Gas usage depends upon the amount of storage and set of instructions (codes) used in a smart contract. The transaction fee is calculated in Ether, which is given as:

Ether = Tx Fees = Gas Limit * Gas Price

where:

- **gasLimit** = maximum amount of gas that is going to spend on a single transaction

- **gasPrice** = minimum gas cost of the execution of a transaction

28. What is the very first thing you must specify in a Solidity file?

It is necessary to specify the version number of Solidity at the beginning of code as it eliminates incompatibility errors that can arise while compiling with another version. This is a mandatory clause that has to be there at the top of any Solidity code you write. You also need to mention the correct version number for the code.

29. What is the use of the payable keyword in Solidity?

The payable keyword in Solidity is used as a modifier for a function or a fallback method to indicate that the function can receive Ether directly. When a function is marked as payable, it enables the contract to accept Ether directly as part of the transaction calling the function.

If a function is not declared as payable, sending Ether to the function will result in a runtime error, causing the transaction to revert.

```
pragma solidity ^0.8.0;

contract PayableExample {
    uint256 public totalReceived;

    function sendEther() public payable {
        totalReceived = totalReceived + msg.value;
    }

    function getBalance() public view returns (uint256) {
        return address(this).balance;
    }
}
```

30. How can you implement a simple contract that accesses the block information in Solidity?

```
pragma solidity ^0.8.0;

contract BlockInfo {

    function getBlockNumber() public view returns (uint256) {
        return block.number;
    }

    function getBlockTimestamp() public view returns (uint256) {
        return block.timestamp;
    }

    function getBlockDifficulty() public view returns (uint256) {
        return block.difficulty;
    }
}
```

31. How can you implement a time lock or delay on function execution in a Solidity contract?

```
pragma solidity ^0.8.0;

contract TimeLock {
    uint256 public unlockTime;
    address public owner;

    modifier onlyOwner {
        require(msg.sender == owner, "Not the owner");
        _;
    }

    modifier unlocked {
        require(block.timestamp >= unlockTime, "Function is locked");
        _;
    }

    constructor(uint256 _lockDuration) {
        owner = msg.sender;
        unlockTime = block.timestamp + _lockDuration;
    }

    function setUnlockTime(uint256 _lockDuration) public onlyOwner {
        unlockTime = block.timestamp + _lockDuration;
    }

    function executeRestrictedFunction() public onlyOwner unlocked {
        // Execute the restricted actions here...
    }
}
```

32. What are function modifiers and provide an example of usage in Solidity?

Function modifiers are a feature in Solidity that allows you to alter the behavior of a function. They can be used to add common requirements, such as access restrictions, preconditions, and state validation checks.

```
pragma solidity ^0.8.0;

contract ModifierExample {
    address public owner;

    constructor() {
        owner = msg.sender;
    }

    // Function modifier which restricts access to the owner
    modifier onlyOwner() {
        require(msg.sender == owner, "Only the owner can call this function");
        _;
    }

    function restrictedFunction() public onlyOwner {
        // Execute function logic...
    }
}
```

33. What is the disadvantage of Constants compared to variables?

The main disadvantage of using constants is that we cannot use any blockchain data.