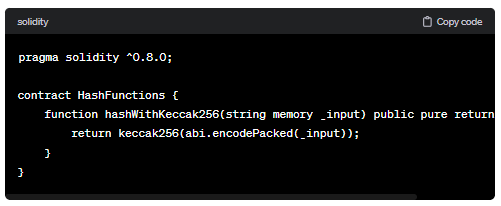
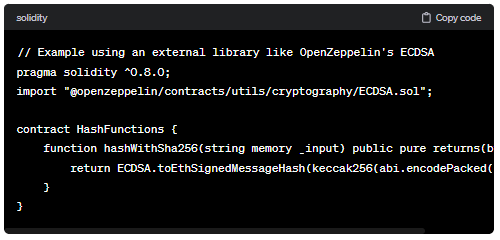
**Inbuilt Cryptographic Hash Function - Keccak256, Sha256, Ripemd160:**

In Solidity, a programming language commonly used for writing smart contracts on the Ethereum blockchain, you can use cryptographic hash functions like Keccak256, SHA256, and RIPEMD160. These functions are typically used for various purposes, including creating unique identifiers, verifying data integrity, and generating hash-based signatures.

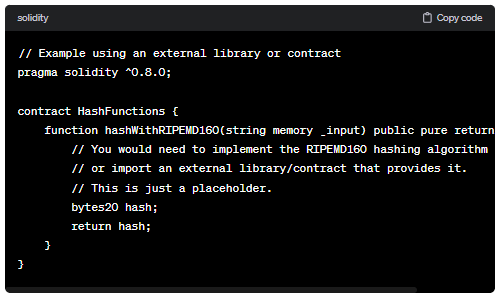
**Keccak256**: This is the hashing function used in Ethereum. You can use it by simply calling **keccak256()**.



**SHA256**: Solidity does not have a built-in SHA256 function, but you can easily integrate it using libraries or by leveraging external contracts that provide this functionality.



**RIPEMD160**: Similar to SHA256, there's no built-in RIPEMD160 function in Solidity, but you can use external libraries or contracts that provide this functionality.



Remember to be cautious when dealing with cryptographic operations in smart contracts, as improper usage can lead to vulnerabilities. Always review and test thoroughly, and consider using well-audited libraries for cryptographic operations.

-> **Keccak256**: It take input as bytes -> give output as hash of 32 bytes. (Different Algo)

-> **Sha256**: It take input as bytes -> give output as hash of 32 bytes. (Different algo)

-> **Ripemd160**: It take input as bytes -> give output as hash of 20 bytes.

Mainly we use “keccak256 and Sha256”.

Hash function is used for:

1) Unique id

2) Contract sign

**Hash collision** -> if same hash value generated for 2 different input values.

We have 2 in-built function -> 1) abi.encode 2) abi.encodepacked

**Code:**

//SPDX-License-Identifier: GPL-3.0

pragma solidity ^0.8.0;

contract hash{

    function hashkeccak256(uint \_x, string memory name, address \_add) public pure returns(bytes32){

        return keccak256(abi.encodePacked(\_x,name,\_add));

    }

    function hashsha256(uint \_x,string memory name, address \_add) public pure returns(bytes32){

        return sha256(abi.encodePacked(\_x,name,\_add));

    }

    function hashRipemd160(uint \_x,string memory name, address \_add) public pure returns(bytes20){

        return ripemd160(abi.encodePacked(\_x,name,\_add));

    }

}

**Output:**

