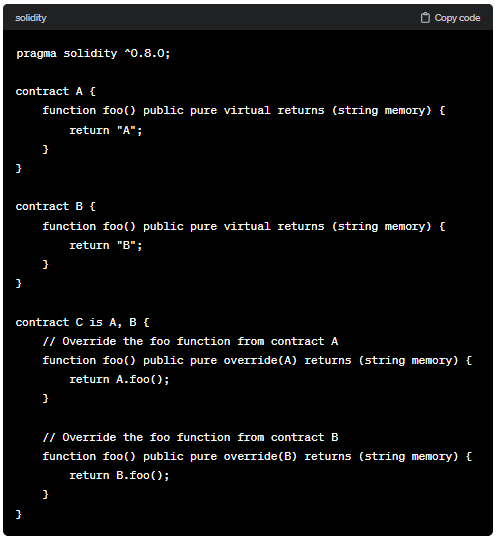
**Overriding Parent Function in Multiple Inheritance:**

In Solidity, when you have multiple inheritance and two or more parent contracts define a function with the same name and signature, you can override the function in the derived contract to specify which parent's implementation should be used. Here's how you can override a parent function in a derived contract with multiple inheritance



By explicitly specifying which parent function to override, you can resolve the ambiguity caused by multiple inheritance and provide specific implementations for each overridden function in the derived contract.

**Code:**

//SPDX-License-Identifier: GPL-3.0

pragma solidity ^0.8.0;

contract A{

    uint public a;

    constructor(){

        a=100;

    }

    function funA() public {

        a=10;

    }

    function fun() public pure virtual returns(string memory){

        return "hi i am in A";

    }

}

contract B is A{

    uint public b;

    constructor(){

        b=200;

        a=50;

    }

    function fun() public pure virtual override returns(string memory){

        return "hi i am in B";

    }

}

contract C is A,B{

    function fun() public pure virtual override(A,B) returns(string memory) {

        return "hi i am in C";

    }

}

contract D is A,B,C{

    function fun() public pure override(A,B,C) returns(string memory) {

        return "hi i am in D";

    }

}

**Output:**



