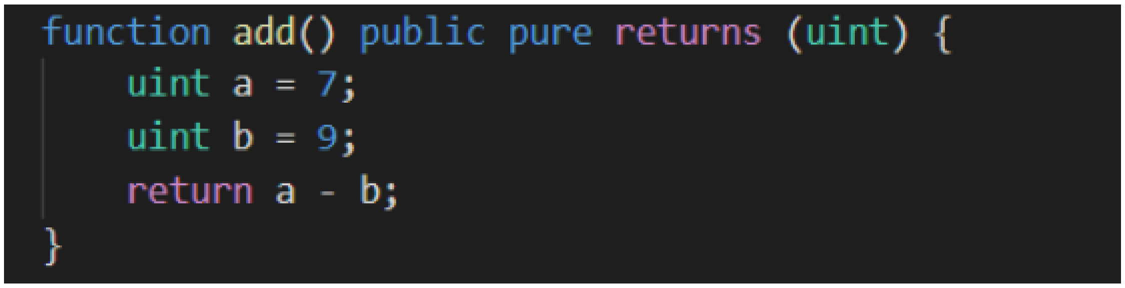
Discord ID: qray#8952
Email: yarooq1@gmail.com

A.



```
function add() public pure returns (uint) {
    uint a = 7;
    uint b = 9;
    return a - b;
}
```

The above function outputs

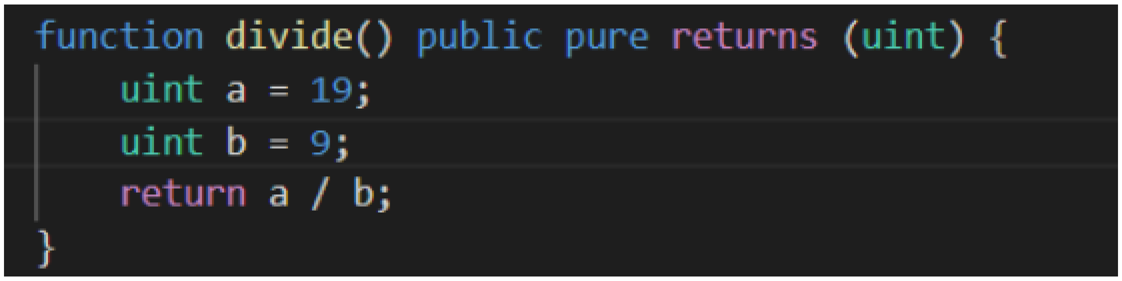
115792089237316195423570985008687907853269984665640564039457584007913129639934

This is because the add() function is expected to return an **unsigned integer** which is a non-negative number but the function returns a negative number i.e (-2) therefore the expected output is not obtained.

Solution

The solution would be using ***int*** data type instead of ***uint*** data type as the return type of the function.

B.



```
function divide() public pure returns (uint) {
    uint a = 19;
    uint b = 9;
    return a / b;
}
```

Solidity has very minimal support for floating point values. So the above divide() function returns a whole number. The reason being that 19 / 9 = 2.111… but solidity only returns 2 and disregards the decimal point values.

Solution

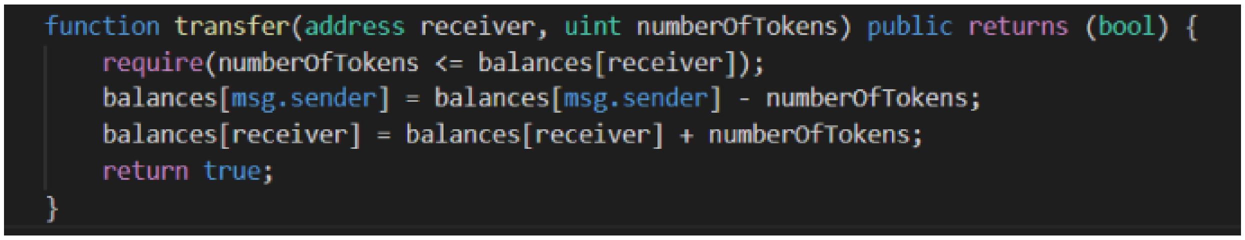
One work around could be that we can multiply the dividend by some multiple of 10 and then divide it by the divisor. In this case

(19 * 1000) / 9 = 2111

By doing this we can get our result as an integer and when need to use it as a decimal point value we could divide it by the same multiple of 10 and then perform the desired operations on it.

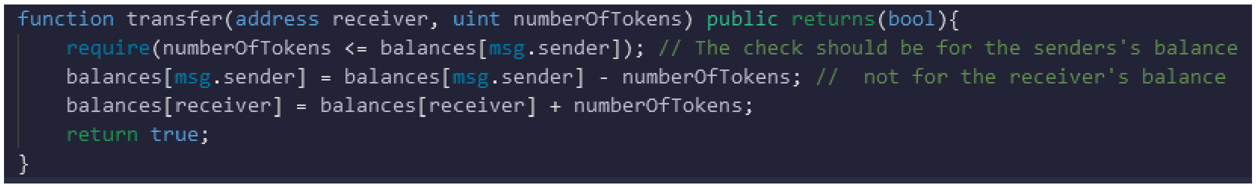
Discord ID: qray#8952
Email: yarooq1@gmail.com

C.



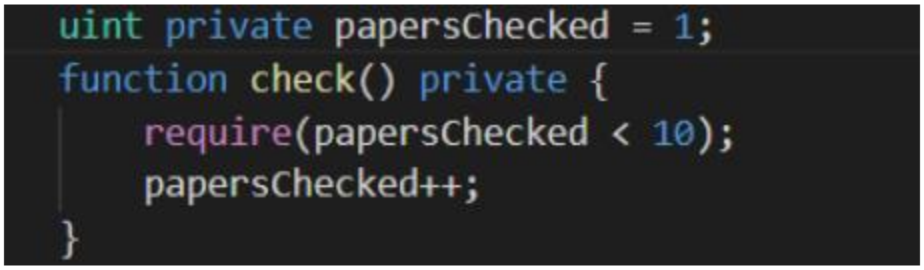
```
function transfer(address receiver, uint numberOfTokens) public returns (bool) {
    require(numberOfTokens <= balances[receiver]);
    balances[msg.sender] = balances[msg.sender] - numberOfTokens;
    balances[receiver] = balances[receiver] + numberOfTokens;
    return true;
}
```

Solution



```
function transfer(address receiver, uint numberOfTokens) public returns(bool){
    require(numberOfTokens <= balances[msg.sender]); // The check should be for the senders's balance
    balances[msg.sender] = balances[msg.sender] - numberOfTokens; //  not for the receiver's balance
    balances[receiver] = balances[receiver] + numberOfTokens;
    return true;
}
```

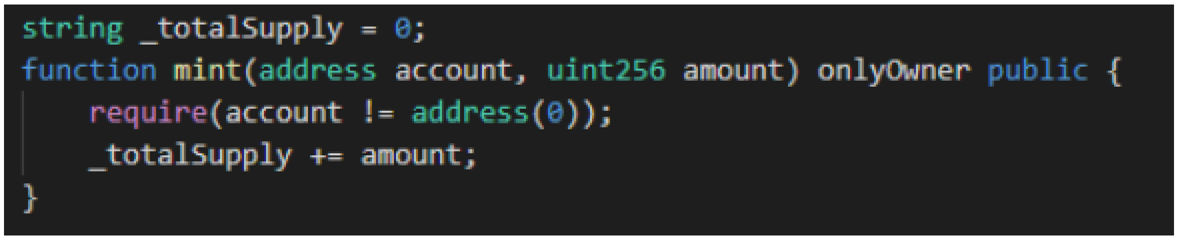
D.



```
uint private papersChecked = 1;
function check() private {
    require(papersChecked < 10);
    papersChecked++;
}
```

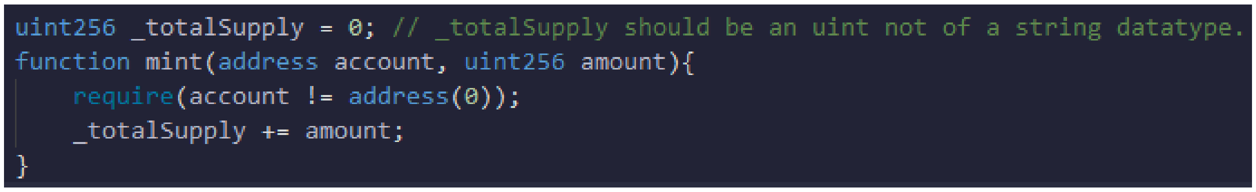
The check() function will update the papersChecked variable **8** more times. Hence the function can be executed **8** more times.

E.



```
string _totalSupply = 0;
function mint(address account, uint256 amount) onlyOwner public {
    require(account != address(0));
    _totalSupply += amount;
}
```

Solution



```
uint256 _totalSupply = 0; // _totalSupply should be an uint not of a string datatype.
function mint(address account, uint256 amount){
    require(account != address(0));
    _totalSupply += amount;
}
```