Solidity的疑难关键字和实现插入排序（易错）

学习网站：https://github.com/AmazingAng/WTF-Solidity

在学习solidity中，遇到了几个难以理解的关键字，分别是：

## 1. pure v.s. view

pure和view是在其他语言中没出现过的关键字。

包含pure关键字的函数，不能读取也不能写入存储在链上的状态变量；

而包含view关键字的函数，能读取但也不能写入状态变量；

不写pure也不写view，函数既可以读取也可以写入状态变量。

pure和view与其他函数关键字在花费gas上的区别：

包含pure跟view关键字的函数是不改写链上状态的，因此用户直接调用他们是不需要付gas的；

合约中非pure/view函数调用它们则会改写链上状态，需要付gas。

### 代码例子：

我们在合约里定义一个状态变量 number = 4。

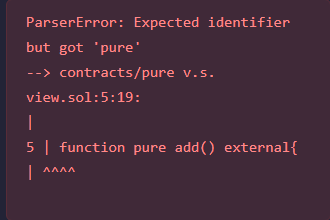
// SPDX-License-Identifier: MIT

pragma solidity ^0.8.4; contract FunctionTypes{ uint256 public number = 4;

定义一个add()函数，每次调用，每次给number + 1。

function add() external{ number = number + 1; }

如果add()包含了pure关键字，例如 function add() pure external，就会报错。



因为pure是不能读取合约里的状态变量的，更不能改写。

同理，如果add()包含view，比如function add() view external，也会报错。因为view能读取，但不能够改写状态变量。

那pure和view函数能做些什么？其实可以稍微改写下方程，让他不改写number，而是返回一个新的变量。

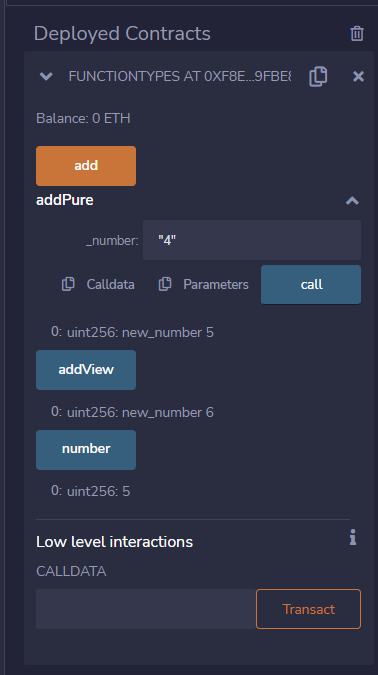
// pure:can't read and write

function addPure(uint256 \_number) external pure returns(uint256 new\_number){ new\_number = \_number + 1; }

// view: can read but can't write

function addView() external view returns(uint256 new\_number) { new\_number = number + 1; }

这样函数就能运行成功了。



## 2.solidity实现插入排序(易错）

// 插入排序（与python代码不同就是把j加1，让它不能取负数）

function insertionSort(uint[] memory a) public pure returns(uint[] memory) {

// note that uint can not take negative value

for (uint i = 1;i < a.length;i++){

uint temp = a[i];

uint j=i;

while( (j >= 1) && (temp < a[j-1])){

a[j] = a[j-1];

j--;

}

a[j] = temp;

}

return(a);

}

用python的思维写上述标红代码容易写为 **uint j = i-1; j>=0; a[j]; a[j] = a[j]**，在python中没有错，但是在solidity中跑会出现问题。因为在solidity中最常用的变量类型是uint，也就是正整数，取到负值的话，会报underflow错误。而在插入算法中，变量j有可能会取到-1，引起报错。上述代码就是我们需要把j加1，让它无法取到负值。

## 总结：

在学习solidity中，不是特别理解pure跟view关键字，在学习过程中也容易忽略这两个关键字的具体含义和其修饰函数对链上状态的改变。solidity中引入pure和view关键字主要是为了节省gas和控制函数权限：含pure和view关键字的函数是不会改变链上状态的，用户调用它们修饰的函数是不会花费gas的；在solidity中最常用的变量类型是uint，就是正整数。希望在此学习总结中希望能帮助到大家具体理解pure跟view这两个关键字。