# 294.https://stackoverflow.com/questions/71754744/erc721a-smart-contract

**T:**ERC721A smart contract

**Q:**I'm writing an ERC721A smart contract and getting a warning in the Remix IDE. There is the code:  
  
function tokenOfOwnerByIndex(address owner, uint256 index) public view override returns (uint256) { if (index >= balanceOf(owner)) revert OwnerIndexOutOfBounds(); uint256 numMintedSoFar = totalSupply(); uint256 tokenIdsIdx; address currOwnershipAddr; // Counter overflow is impossible as the loop breaks when uint256 i is equal to another uint256 numMintedSoFar. unchecked { for (uint256 i; i < numMintedSoFar; i++) { TokenOwnership memory ownership = \_ownerships[i]; if (ownership.addr != address(0)) { currOwnershipAddr = ownership.addr; } if (currOwnershipAddr == owner) { if (tokenIdsIdx == index) { return i; } tokenIdsIdx++; } } } // Execution should never reach this point. assert(false);}  
  
WARN: THIS PARAGRAPH CONTAINS TAG: [CODE]   
  
This code is from the Square Bears collection ( https://etherscan.io/address/0x2b1037def2aa4ed427627903bdef9bdd27ae1ea3#code ). I got it from a YouTube tutorial. I think the code works, but I keep getting a warning.  
  
Warning: Unnamed return variable can remain unassigned. Add an explicit return with value to all non-reverting code paths or name the variable.--> contracts/ERC721A.sol:103:94:|103 | function tokenOfOwnerByIndex(address owner, uint256 index) public view override returns (uint256) {| ^^^^^^^  
  
WARN: THIS PARAGRAPH CONTAINS TAG: [CODE]   
  
I assume that I have to provide a named return value or variable, but the code seems to return an iterated value (i).

**C1:**The unchecked code looks sketchy why are you using something out of the box like OpenZeppelins

**C2:**Could you explain more about unchecked code. I took this code from a verified contract on etherscan

**C3:**Open Zepplin is your friend wizard.openzeppelin.com/#erc721

1 **Answer**

**A1:**Because you've told the compiler that you will return a value from the function, but you didn't.  
  
You should return a mock value even if you don't need to.  
  
 ... // Execution should never reach this point. assert(false); return tokenIdsIdx; // or simply return 0;}  
  
WARN: THIS PARAGRAPH CONTAINS TAG: [CODE]