# 603.https://stackoverflow.com/questions/69408191/erc-721-smart-contract-is-minting-2-nfts-at-a-time

**T:**ERC-721 Smart contract is minting 2 NFTs at a time

**Q:**I have this smart contract from Hash Lip's github that, from what I can tell should be minting 1 at a time, but is instead minting 2 every time. Code as follows:  
  
Setup code:  
  
// SPDX-License-Identifier: GPL-3.0// Created by HashLips// The Nerdy Coder Clonespragma solidity ^0.8.0;import "@openzeppelin/contracts/token/ERC721/extensions/ERC721Enumerable.sol";import "@openzeppelin/contracts/access/Ownable.sol";contract TestBoxes is ERC721Enumerable, Ownable { using Strings for uint256; string public baseURI; string public baseExtension = ".json"; uint256 public cost = 0.01 ether; uint256 public maxSupply = 3; //there should only be 3 minted (I have 3 image files to test) uint256 public maxMintAmount = 3; bool public paused = false; mapping(address => bool) public whitelisted;  
  
WARN: THIS PARAGRAPH CONTAINS TAG: [CODE]   
  
And then the part where the contract is minting is as follows. as you can see above, I have set the max as 3, and in the next part, after the constructor executes, it mints 1 NFT for the owner.  
  
 constructor( string memory \_name, string memory \_symbol, string memory \_initBaseURI ) ERC721(\_name, \_symbol) { setBaseURI(\_initBaseURI); mint(msg.sender, 1); //should mint 1 at deployment but mints 2... } // internal function \_baseURI() internal view virtual override returns (string memory) { return baseURI; } // public function mint(address \_to, uint256 \_mintAmount) public payable { uint256 supply = totalSupply(); require(!paused); require(\_mintAmount > 0); require(\_mintAmount <= maxMintAmount); require(supply + \_mintAmount <= maxSupply); if (msg.sender != owner()) { if(whitelisted[msg.sender] != true) { require(msg.value >= cost \* \_mintAmount); } } for (uint256 i = 0; i <= \_mintAmount; i++) { //start the index at 0 \_safeMint(\_to, supply + i); } }  
  
WARN: THIS PARAGRAPH CONTAINS TAG: [CODE]

**C1:**I don't want to say anything bad about hashlips but you should probably find a different source for your smart contract code.

2 **Answer**

**A1:**You have a logical error in the for loop within the mint() function.  
  
WARN: THIS PARAGRAPH CONTAINS TAG: [CODE]   
  
for (uint256 i = 0; i <= \_mintAmount; i++)  
  
WARN: THIS PARAGRAPH CONTAINS TAG: [CODE]   
  
Example: \_mintAmount is 1 (same as passing from the constructor).  
  
WARN: THIS PARAGRAPH CONTAINS TAG: [CODE]   
  
 ● First iteration:  
i is 0, which is less or equal 1  
=> performs the iteration and executes the \_safeMint()  
  
  
 ● Second iteration:  
i is 1, which is less or equal 1  
=> it still performs the iteration and executes the \_safeMint()  
  
  
Fix it by changing the condition to i < \_mintAmoun (i is less than). Then it will only execute once (for mintAmount value 1).  
  
WARN: THIS PARAGRAPH CONTAINS TAG: [CODE]

**A2:**for (uint256 i = 1; i <= \_mintAmount; i++)  
  
WARN: THIS PARAGRAPH CONTAINS TAG: [CODE]   
  
The way above works the same way.