zombieownership.sol

```
pragma solidity ^0.4.25;

import "./zombieattack.sol";
import "./erc721.sol";
import "./safemath.sol";

contract ZombieOwnership is ZombieAttack, ERC721 {

  using SafeMath for uint256;

  mapping (uint => address) zombieApprovals;

  function balanceOf(address _owner) external view returns (uint256) {
    return ownerZombieCount[_owner];
  }

  function ownerOf(uint256 _tokenId) external view returns (address) {
    return zombieToOwner[_tokenId];
  }

  function _transfer(address _from, address _to, uint256 _tokenId) private {
    ownerZombieCount[_to] = ownerZombieCount[_to].add(1);
    ownerZombieCount[msg.sender] = ownerZombieCount[msg.sender].sub(1);
    zombieToOwner[_tokenId] = _to;
    emit Transfer(_from, _to, _tokenId);
  }

  function transferFrom(address _from, address _to, uint256 _tokenId) external
payable {
    require (zombieToOwner[_tokenId] == msg.sender ||
zombieApprovals[_tokenId] == msg.sender);
    _transfer(_from, _to, _tokenId);
  }

  function approve(address _approved, uint256 _tokenId) external payable
onlyOwnerOf(_tokenId) {
    zombieApprovals[_tokenId] = _approved;
    emit Approval(msg.sender, _approved, _tokenId);
  }

}
```