zombiehelper.sol

```
pragma solidity ^0.4.25;

import "./zombiefeeding.sol";

contract ZombieHelper is ZombieFeeding {

  uint levelUpFee = 0.001 ether;

  modifier aboveLevel(uint _level, uint _zombieId) {
    require(zombies[_zombieId].level >= _level);
    _;
  }

  function withdraw() external onlyOwner {
    address _owner = owner();
    _owner.transfer(address(this).balance);
  }

  function setLevelUpFee(uint _fee) external onlyOwner {
    levelUpFee = _fee;
  }

  function levelUp(uint _zombieId) external payable {
    require(msg.value == levelUpFee);
    zombies[_zombieId].level = zombies[_zombieId].level.add(1);
  }

  function changeName(uint _zombieId, string _newName) external
aboveLevel(2, _zombieId) onlyOwnerOf(_zombieId) {
    zombies[_zombieId].name = _newName;
  }

  function changeDna(uint _zombieId, uint _newDna) external aboveLevel(20,
_zombieId) onlyOwnerOf(_zombieId) {
    zombies[_zombieId].dna = _newDna;
  }

  function getZombiesByOwner(address _owner) external view returns(uint[]) {
    uint[] memory result = new uint[](ownerZombieCount[_owner]);
    uint counter = 0;
    for (uint i = 0; i < zombies.length; i++) {
      if (zombieToOwner[i] == _owner) {
```

```
        result[counter] = i;
        counter++;
      }
    }
    return result;
  }

}
```